НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

Теплоенергетичний факультет

Кафедра автоматизації проектування енергетичних процесів і систем

"На правах рукопису"
УДК 004.422.833

«До захисту допущено»
Завідувач кафедри
Коваль О.В. _____
(прізвище, ініціали) (підпис)
«____» _____ 2018р.

Магістерська дисертація

зі спеціальності - 121 Інженерія програмного забезпечення
за спеціалізацією - Програмне забезпечення розподілених систем
на тему: «Комп'ютерна безпека в системах моделювання гідроакустичних процесів»

Виконав: студент VI курсу, групи ТВ-71мп
___Тобілко Андрій Олександрович_____ _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник _к.т.н. доцент Кублій Л.І._____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Рецензент_____ _____
(посада, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Київ - 2018

Національний технічний університет України
“ Київський політехнічний інститут ім. Ігоря Сікорського”

Факультет теплоенергетичний

Кафедра автоматизації проектування енергетичних процесів і систем

Рівень вищої освіти другий, магістерський

зі спеціальності - 121 Інженерія програмного забезпечення
за спеціалізацією - Програмне забезпечення розподілених систем

ЗАТВЕРДЖУЮ
Завідувач кафедри
Коваль О.В. _____
(прізвище, ініціали) (підпис)
«____» _____ 2018р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ

Тобілко Андрію Олександровичу
(прізвище, ім’я, по батькові)

1. Тема дисертації Комп’ютерна безпека в системах моделювання гідроакустичних процесів
Науковий керівник Кублій Лариса Іванівна, к.т.н., доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)
затверджені наказом по університету від 5 листопада 2018 року №4072-с
2. Строк подання студентом дисертації 13 грудня 2018
3. Об’єкт дослідження _захист систем моделювання гідроакустичних процесів
4. Предмет дослідження механізми гарантування безпеки систем моделювання гідроакустичних процесів
5. Перелік питань, які потрібно розробити проаналізувати існуючі системи моделювання гідроакустичних процесів; проаналізувати проблеми безпеки в даних системах; абстрагувати найважливіші проблеми захисту; знайти рішення даних проблем; створити програмне рішення для забезпечення захисту відповідно до виокремлених проблем; створити тестову систему моделювання гідроакустичних процесів; протестувати програмне рішення на тестові системі моделювання
6. Перелік ілюстративного матеріалу мікросервісна система, засоби автоматичного масштабування, функції програмного забезпечення, структура програмного забезпечення, інтерфейс

7. Орієнтований перелік публікацій _____
1) Кублій Л.І., Тобілко А.О. “Комп'ютерна безпека в комплексі моделювання гідроакустичних процесів”
2) Кублій Л.І., Тобілко А.О. “Захист інформації в комплексі моделювання гідроакустичних процесів”

8. Дата видачі завдання «_11_»__вересня____ 2017 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз існуючих систем моделювання гідроакустичних процесів	11.09.2017-12.11.2017	
2	Аналіз проблем в безпеці систем моделювання	13.11.2017-15.01.2018	
3	Виокремлення найважливіших проблем в безпеці систем моделювання	16.01.2018-05.03.2018	
4	Аналіз рішень для усунення проблем безпеки систем моделювання	06.03.2018-27.06.2018	
5	Аналіз інструментів для реалізації рішень щодо усунення проблем безпеки	28.06.2018-18.08.2018	
6	Моделювання схеми роботи майбутньої програми	19.08.2018-30.08.2018	
7	Розробка архітектури програмного забезпечення	01.09.2018-20.09.2018	
8	Розробка програмного продукту	21.09.2018-10.10.2018	
9	Розробка демонстраційного додатку	11.10.2018-30.11.2018	
10	Оформлення документації	01.12.2018-07.12.2018	

Студент _____ __Тобілко А.О._____
(підпис) (прізвище та ініціали)

Науковий керівник _____ __Кублій Л.І._____
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Магістерська дисертація складається зі вступу, чотирьох розділів, висновку, переліку посилань з 50 найменувань, 2 додатків, і містить 40 рисунків, 22 таблиці. Повний обсяг магістерської дисертації складає 113 сторінок, з яких перелік посилань займає 4 сторінок, додатки – 11 сторінок.

Зважаючи на неймовірно стрімкий розвиток гідроакустичної галузі, збільшення кількості досліджень, що стосуються моделювання та аналізу гідроакустичних процесів, питання надання захисту системам, що працюють в гідроакустичному середовищі є дуже актуальним.

Досліджуючи основні рішення для захисту систем моделювання гідроакустичних процесів, можна стверджувати, що на ринку відсутній універсальний програмний продукт, що може вирішити основні питання безпеки гідроакустичної системи.

Типова система моделювання гідроакустичних процесів повинна була захищена на основі загальних принципів комп'ютерної безпеки, що можна описати наступними рівнями:

- рівень автентифікації;
- рівень авторизації;
- рівень шифрування даних;
- рівень балансування навантаження;
- рівень стиснення даних.

Крім загальних проблем комп'ютерної безпеки, гостро постають питання знаходження інших важливих проблем безпеки типової системи моделювання гідроакустичних процесів, ретельного дослідження цих проблем, та їх аналізу. Аналіз найважливіших проблем в системі безпеки повинен включати огляд існуючих рішень та їх ефективність.

Основними завданнями дослідження є: проаналізувати велику кількість систем моделювання гідроакустичних процесів; дослідити загальні та найважливіші прогалини в безпеці досліджуваних систем; детально переглянути існуючі рішення,

що пропонують засоби для вирішення цих проблем, та визначити їх універсальність та ефективність. На основі результатів, отриманих на основі вирішення поставлених вище питань, створити систему, що

- гарантує безпеку системи моделювання гідроакустичних процесів;
- надає вищезгадані рівні захисту, з можливістю детального налаштування кожного з рівнів;
- є адаптивним програмним рішенням, тобто таким, що може застосованим до широкої вибірки систем моделювання гідроакустичних процесів.

Важливим етапом магістерської роботи має бути створення тестової системи моделювання гідроакустичних процесів. Для демонстрації роботи системи захисту також має бути створений клієнт системи моделювання з простим та зрозумілим інтерфейсом користувача.

Практична цінність магістерської дисертації визначається тим, що надання базових та універсальних засобів для забезпечення комп'ютерної безпеки дає можливість їх інтеграції до будь-якої системи моделювання гідроакустичних процесів.

Робота виконана в межах науково-дослідницького напряму роботи кафедри автоматизації проектування енергетичних процесів і систем теплоенергетичного факультету Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Результати роботи прийняті та інтегровані в межах дослідницького центру кафедри.

Основні результати атестаційної роботи обговорювалися на:

- XVI міжнародна науково-практична конференція аспірантів, магістрантів, студентів «Сучасні проблему наукового забезпечення енергетики» (24-27 квітня 2018 року).

За матеріалами роботи опубліковано 2 наукові роботи.

Результати роботи впроваджено у ТОВ «НВП «Символ».

Ключові слова: *КОМП'ЮТЕРНА БЕЗПЕКА, ЗАХИСТ СИСТЕМИ, СИСТЕМА МОДЕЛЮВАННЯ ГІДРОАКУСТИЧНИХ ПРОЦЕСІВ*

ABSTRACT

The master's dissertation consists of an introduction, four sections, a conclusion, a list of references from 50 names, 2 applications, and contains 40 figures, 22 tables. The full volume of the master's thesis consists of 113 pages, of which the list of links takes 4 pages, applications 11 pages.

In view of the incredibly rapid development of the hydroacoustic industry, the increasing number of studies relating to the modeling and analysis of hydroacoustic processes, the issue of providing protection to systems operating in the hydroacoustic environment is very relevant.

Investigating the main solutions for the protection of hydroacoustic modeling systems, one can state that there is no universal software product that can solve the basic issues of the safety of the hydroacoustic system.

A typical hydroacoustic process modeling system was protected on the basis of the general principles of computer security, which can be described by the following levels:

- authentication level;
- authorization level;
- data encryption level;
- load balancing level;
- data compression level.

In addition to the general problems of computer security, the problem of finding other important security problems of the typical model of hydroacoustic processes, the careful study of these problems and their analysis are acute. The analysis of critical security issues should include an overview of existing solutions and their effectiveness.

The main tasks of the research are: to analyze a large number of systems of modeling hydroacoustic processes; to investigate the general and critical gaps in the safety of the systems being studied; to review in detail the existing solutions offering remedies for these problems and to determine their versatility and effectiveness. Based on the results obtained on the basis of the solution of the above questions, create a system that

- guarantees the safety of the hydroacoustic process simulation system;

- Provides the above-mentioned security levels, with the possibility of detailed adjustment of each level;
- is an adaptive software solution, that is, such that it can be applied to a wide selection of simulation systems of hydroacoustic processes.

An important stage of master's work should be the creation of a test system for modeling hydroacoustic processes. To demonstrate the work of the security system, a simulation client must be created with a simple and intuitive user interface.

The practical value of the master's thesis is determined by the fact that the provision of basic and universal means for ensuring computer security enables them to integrate into any system of modeling of hydroacoustic processes.

The work was carried out within the framework of the research direction of the Department of Automation of the Design of Energy Processes and Systems of the Heat and Power Faculty of the National Technical University of Ukraine "Kyiv Polytechnic Institute named after Igor Sikorsky". The results of work are accepted and integrated within the research center of the department.

The main results of attestation work were discussed at:

- XVI International Scientific and Practical Conference of Postgraduates, Graduates, Students "Modern Problems of Scientific Supply of Power Engineering" (April 24-27, 2018).

On the basis of the work published 2 scientific works.

The results of the work were implemented at the "Scientific and production enterprise" Symbol "LLC.

Keywords: COMPUTER SECURITY, SYSTEM PROTECTION, SYSTEM OF MODELING OF HYDRAACOUS PROCESSES

ЗМІСТ

ВСТУП

Забезпечення комп’ютерної безпеки в системах моделювання гідроакустичних процесів має велике значення як для науковців, що працюють в сфері аналізу та моделювання гідроакустичних процесів, так і для провідних спеціалістів з питань комп’ютерної безпеки [1].

Зважаючи на неймовірно стрімкий розвиток гідроакустичної галузі, збільшення кількості досліджень, що стосуються моделювання та аналізу гідроакустичних процесів, питання надання захисту системам, що працюють в гідроакустичному середовищі є дуже актуальним.

Досліджуючи основні рішення для захисту систем моделювання гідроакустичних процесів, можна стверджувати, що на ринку відсутній універсальний програмний продукт, що може вирішити основні питання безпеки гідроакустичної системи.

Типова система моделювання гідроакустичних процесів повинна була захищена на основі загальних принципів комп’ютерної безпеки, що можна описати наступними рівнями:

- рівень автентифікації;
- рівень авторизації;
- рівень шифрування даних;
- рівень балансування навантаження;
- рівень стиснення даних.

Крім загальних проблем комп’ютерної безпеки, гостро постають питання знаходження інших важливих проблем безпеки типової системи моделювання гідроакустичних процесів, ретельного дослідження цих проблем, та їх аналізу. Аналіз найважливіших проблем в системі безпеки повинен включати огляд існуючих рішень та їх ефективність.

Основними завданнями дослідження є: проаналізувати велику кількість систем моделювання гідроакустичних процесів; дослідити загальні та найважливіші прогалини в безпеці досліджуваних систем; детально переглянути існуючі рішення,

що пропонують засоби для вирішення цих проблем, та визначити їх універсальність та ефективність. На основі результатів, отриманих на основі вирішення поставлених вище питань, створити систему, що

- гарантує безпеку системи моделювання гідроакустичних процесів;
- надає вищезгадані рівні захисту, з можливістю детального налаштування кожного з рівнів;
- є адаптивним програмним рішенням, тобто таким, що може застосованим до широкої вибірки систем моделювання гідроакустичних процесів.

Важливим етапом магістерської роботи має бути створення тестової системи моделювання гідроакустичних процесів. Для демонстрації роботи системи захисту також має бути створений клієнт системи моделювання з простим та зрозумілим інтерфейсом користувача [2].

Практична цінність магістерської дисертації визначається тим, що надання базових та універсальних засобів для забезпечення комп’ютерної безпеки дає можливість їх інтеграції до будь-якої системи моделювання гідроакустичних процесів.

Робота виконана в межах науково-дослідницького напряму роботи кафедри автоматизації проектування енергетичних процесів і систем теплоенергетичного факультету Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Результати роботи прийняті та інтегровані в межах дослідницького центру кафедри. За результатами дипломної роботи було здійснено доповіді на конференціях і опубліковано наукову статтю.

1. АНАЛІЗ ПРОБЛЕМ КОМП'ЮТЕРНОЇ БЕЗПЕКИ В СИСТЕМАХ МОДЕЛЮВАННЯ ГІДРОАКУСТИЧНИХ ПРОЦЕСІВ

Комп'ютерна безпека в системах моделювання гідроакустичних процесів потребує детального дослідження. Першим кроком даного дослідження повинний бути аналіз загальних та добре відомих проблем безпеки. Науковці та спеціалісти з комп'ютерної безпеки виділяють наступні важливі рівні [3][4]:

- рівень автентифікації;
- рівень авторизації;
- рівень шифрування даних;
- рівень балансування навантаження;
- рівень стиснення даних.

Наступним кроком повинно бути дослідження менш відомих, але не менш серйозних проблем в межах згаданих вище рівнів.

Даний розділ поєднує опис цих двох рівнів, аналізує всі проблеми, що можуть виникнути в системах моделювання гідроакустичних процесів, та підсумовує їх.

1.1 Аналіз проблем рівня автентифікації

На перший погляд, автентифікація концептуально є найпростішої з усіх механізмів безпеки, які використовуються в програмних продуктах. Зазвичай, користувач надає своє ім'я користувача та пароль, а додаток повинен перевірити правильність цих елементів. Якщо дані коректні, автентифікація дозволяє користувачеві ввійти в систему. Якщо ні – заборона входу [5].

Автентифікація також лежить в основі захисту програми від шкідливих атак. Це перша лінія захисту від несанкціонованого доступу. Якщо зловмисник може перемогти ці захисні сили, він часто отримає повний контроль над функціональними можливостями програми та необмежений доступ до даних, що зберігаються в системі.

Без надійної автентифікації, на яку можна покладатися, жоден з інших основних механізмів безпеки (наприклад, керування сеансами та контроль доступу) не може бути ефективним.

Фактично, незважаючи на свою очевидну простоту, розробка функції безпечної автентифікації є дуже важливим питанням. В реальних мережах автентифікація веб-програм часто є найслабшою ланкою, яка дозволяє зловмиснику отримати несанкціонований доступ.

Перш ніж детально розглянути широке розмаїття недоліків дизайну та реалізації, варто познайомитися з технологіями автентифікації.

1.1.1 Технології автентифікації

Під час впровадження механізмів автентифікації розробникам веб-додатків доступний широкий спектр технологій:

- HTML-форматування на основі автентифікації багатофакторних механізмів, таких як об'єднання паролів та фізичних токенів;
- сертифікати клієнта SSL та / або смарт-картки;
- HTTP автентифікація;
- Windows-інтегрована автентифікація за допомогою NTLM або Kerberos;
- сертифікати автентифікації.

Безумовно, найпоширеніший механізм автентифікації, в веб-середовищі, використовує HTML-форми для отримання імені та пароля користувача та їх подання до програми. Цей механізм налічує більше 90% програм, які ви, ймовірно, зустрінете в Інтернеті.

У більш критично важливих для Інтернету додатках, таких як банкові онлайн системи, цей базовий механізм часто розповсюджується на кілька етапів, що вимагає від користувача подання додаткових облікових даних, таких як PIN-код або вибрані символи з секретного слова. Форми HTML, як правило, використовуються для захоплення відповідних даних [6].

У найбільш критично важливих програмах, таких як приватні банківські послуги для високооплачуваних осіб, часто зустрічаються багатофакторні механізми, що використовують фізичні токени.

Ці токени зазвичай створюють потік одноразових кодів доступу або виконують функцію відповіді на виклик на основі введення, вказаного програмою.

Оскільки вартість цієї технології з часом зменшується, скоріш за все, більша кількість додатків використовуватиме такий механізм.

Однак багато хто з цих рішень насправді не вирішують загрози, для яких вони були розроблені - насамперед, атаки, що використовують троянські програми на стороні клієнта.

Деякі веб-додатки використовують SSL-сертифікати на стороні клієнта або криптографічні механізми, реалізовані в рамках смарт-карт.

Через накладні витрати на адміністрування та розподіл цих елементів вони, як правило, використовуються лише в критично важливих контекстах, де невелика користувацька база програми, наприклад веб-VPN для віддалених офісних працівників.

Механізми автентифікації на базі HTTP рідко використовуються в Інтернеті. Вони набагато частіше зустрічаються в середовищах внутрішньої мережі, де внутрішні користувачі організації отримують доступ до корпоративних додатків, надаючи свої звичайні облікові дані мережі або домену. Заявка потім обробляє ці облікові дані за допомогою однієї з цих технологій [7].

Служби сторонніх служб автентифікації, наприклад, паспорт Microsoft, іноді зустрічаються, але наразі вони не були прийняті в жодному значному масштабі. Більшість проблем та атак, які виникають у зв'язку з автентифікацією, можуть застосовуватися до будь-якої з згаданих технологій.

Через перевагу автентифікації на основі HTML-форм потрібно описати кожну конкретну вразливість та атаку в цьому контексті. У відповідних випадках ми будемо вказувати на будь-які специфічні відмінності та методології нападу, які мають відношення до інших доступних технологій.

1.1.2 Недоліки дизайну механізмів автентифікації

Функції автентифікації часто є найслабшою ланкою системи безпеки, ніж будь-який інший механізм безпеки, який зазвичай використовується у веб-додатках.

Навіть у очевидно простій, стандартній моделі, де програма автентифікації користувачів на основі імені та паролю користувача, недоліки в дизайні цієї моделі можуть залишити програму дуже вразливою до несанкціонованого доступу.

1.1.2.1 Погані паролі

Багато веб-додатків не використовують або мінімально контролюють якість паролів користувачів. Зазвичай зустрічаються програми, які дозволяють використовувати паролі:

- дуже короткі або порожні паролі;
- загальні словарні слова або імена;
- паролі, що відповідають імені користувача;
- паролі, налаштовані по замовчуванню.

На рисунку 1.1 наведено приклад слабких правил якості пароля. Кінцеві користувачі зазвичай демонструють мало обізнаності про проблеми безпеки. Зловмисник може легко вгадати ці паролі облікових записів, надаючи йому або несанкціонований доступ до програми [8].

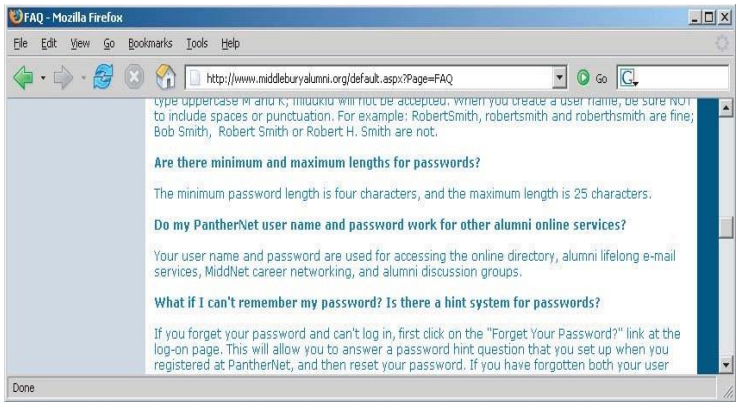

Рисунок 1.1 – Приклад слабких вимог якості паролю

1.1.2.2 Брутальний вхід

Функція входу в систему представляє відкрите запрошення для зловмисника спробувати вгадати ім’я і пароль користувача, а отже, отримати неавторизований доступ до програми. Якщо програма дозволяє зловмисникові робити повторні спроби входу в систему, наприклад «Повторіть спробу за допомогою іншого паролю", поки він не визначить правильним, система є дуже вразливою для атакуючої сторони, який вручну вводить деякі загальні імена користувачів і паролі в свій браузер.

Недавні атаки на високо-профільні сайти забезпечили доступ до сотень тисяч справжніх паролів, які зберігалися як у вигляді чистого тексту, так і з використанням насильницьких хеш-значень [9].

Ось найпопулярніші паролі реального часу:

- password;
- website name;
- 12345678;
- qwerty;
- abc123;
- 111111;
- monkey;
- 12345;
- letmein.

Варіант попередньої уразливості виникає, коли невдалій логін утримується протягом поточного сеансу. Все, що повинен зробити атакуюча сторона, це отримати новий сеанс (наприклад, призупинити його сеансовий файл cookie), і він може продовжити атаку на вгадування паролем [10].

Нарешті, в деяких випадках програма блокує цільовий обліковий запис після відповідної кількості невдалих логінів. Однак він реагує на додаткові спроби входу з повідомленнями, які вказують (або дозволяють зловмиснику зробити висновок), чи правильно вказаний пароль. Це означає, що зловмисник може завершити свій атакуючий пароль, навіть якщо цільовий обліковий запис заблоковано. Якщо

програма автоматично розблоковує облікові записи після певної затримки, зловмисник просто повинен чекати, поки це станеться, а потім ввійти як завжди з відкритим паролем.

На рисунку 1.2 зображена успішна атака по підбору паролю.

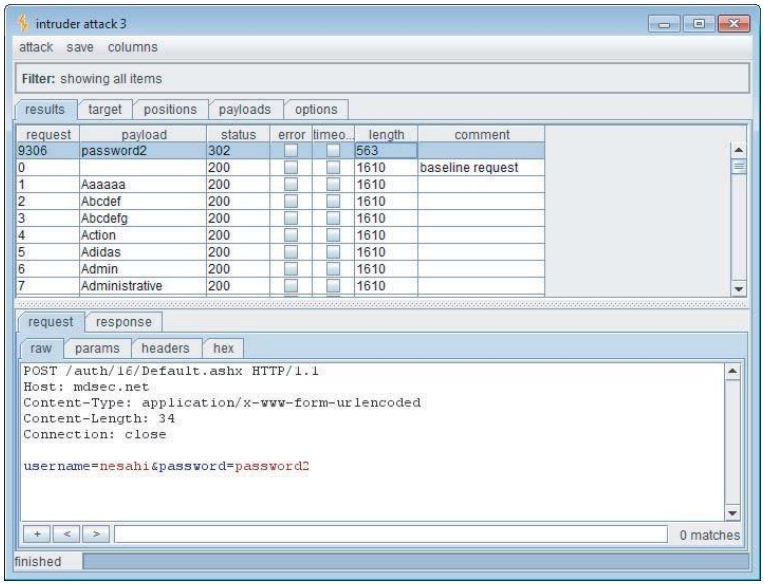

Рисунок 1.2 – Успішна атака по підбору паролю

1.1.2.3 Вербальні повідомлення про помилку

Типова форма входу вимагає від користувача введення двох частин інформації: ім'я та пароль користувача. Для деяких додатків потрібно ввести ще кілька полів, наприклад, дати народження, пам'ятного місця або PIN-код [11].

Коли спроба реєстрації не вдається, ви, звичайно, можете зробити висновок, що принаймні одна частина інформації була невірною. Однак, якщо програма повідомить вам, яка інформація була недійсною, ви можете використовувати цю поведінку, щоб значно зменшити ефективність механізму входу [12].

У найпростішому випадку, коли для входу в систему потрібне ім'я користувача та пароль, програма може відповісти на невдалу спробу входу, вказавши, що причиною невдачі є невизнане ім'я користувача або неправильний пароль, як це показано на рисунку 1.3.

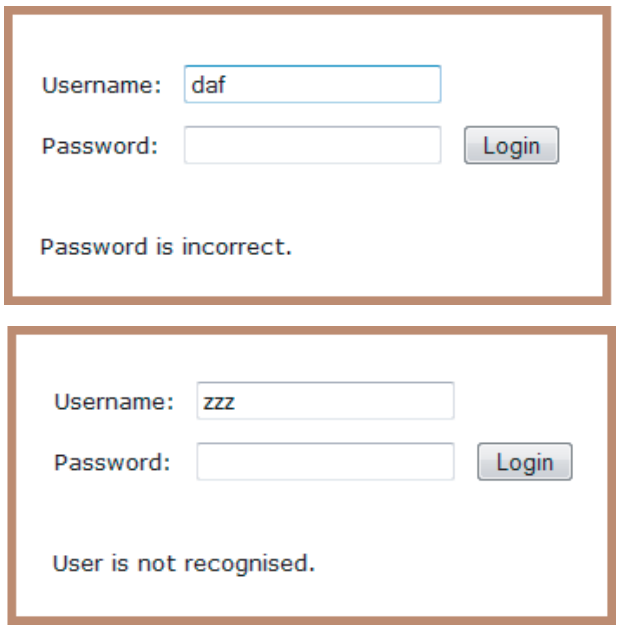

Рисунок 1.3 – Приклад вербального повідомлення про помилку

У цьому випадку ви можете використовувати автоматичну атаку для ітерації за допомогою великого списку загальних імен користувачів, щоб визначити, які з них дійсні. Звичайно, імена користувачів зазвичай не вважаються таємницею (вони, наприклад, не маскуються під час входу в систему). Проте, надаючи легкий засіб для злочинця визначити дійсні імена користувачів, збільшується вірогідність того, що він зможе завдати шкоди програмі перейшовши до наступного кроку: вгадування паролю для конкретного користувача. Список перерахованих імен користувачів може бути використаний як основа для різних наступних атак, включаючи вгадування паролю, атаки на дані користувача або сеанси або соціальну інженерію [13].

Окрім основної функції входу, перелік імен користувачів може виникати в інших компонентах механізму автентифікації. У принципі, для цієї мети може використовуватися будь-яка функція, в якій надається фактичне чи потенційне ім'я користувача. Одне місце, де переліт імені користувача зазвичай зустрічається у функції реєстрації користувача. Якщо додаток дозволяє новим користувачів реєструватися та вказувати власні імена користувачів, то перебір імен користувачів практично неможливо запобігти, якщо програма не дозволяє зареєструвати дубльовані імена користувачів. В інших місцях, де за іменем користувача іноді проводиться зміна пароля та функція забутого пароля, можуть виникнути проблеми.

1.1.2.4 Вразлива передача даних

Якщо програма використовує незашифроване HTTP-з'єднання для передачі критичних даних для входу, підслухування, яке може розташовуватися в мережі, може, звичайно, перехоплювати їх [14]. Залежно від місцезнаходження користувача, потенційні підслухувачі можуть перебувати:

- в локальній мережі користувача;
- в IT-відділі користувача;
- в межах інтернет-провайдера користувача;
- в межах магістральній мережі Інтернету;
- в межах інтернет-провайдера, який розміщує заявку;
- в межах відділу інформаційних технологій, що керує програмою.

Навіть якщо вхід через HTTPS відбувається, облікові дані все ще можуть бути розкриті незахищеним сторонам, якщо програма обробляє їх небезпечно.

Якщо облікові дані передаються як параметри рядка запиту, а не в тілі запиту POST, вони можуть бути зареєстровані в різних місцях, наприклад, в історії веб-переглядача користувача, в журналах веб-сервера та в журналах будь-яких зворотніх проксі-серверів, що використовуються в хостинг-інфраструктурах. Якщо зловмиснику вдалося скомпрометувати будь-якого з цих ресурсів, він може мати можливість перерозподілити привілеї, зафіксувавши в ньому дані облікові дані користувача [15].

Хоча більшість веб-програм використовують тіло запиту POST для подання самої форми реєстрації HTML, дуже часто видно і зрозуміло, що запит на вхід використовується за допомогою перенаправлення на іншу URL-адресу з тими ж обліковими даними, що передаються як параметри рядка запиту. Чому розробники програми вважають необхідним виконувати ці дії незрозуміло, але, вибравши це, простіше реалізовувати операцію як HTTP 302 переспрямування на URL-адресу, ніж запит POST за допомогою другої HTML-форми, поданої через JavaScript.

Веб-додатки іноді зберігають облікові дані користувачів в файлах cookie, як правило, для реалізації погано розроблених механізмів для входу в систему, зміни

пароля, функції "Запам'ятати мене" тощо. Ці облікові дані є вразливими для захоплення через атаки, які погіршують користувацькі cookie файли та, у випадку постійних файлів cookie, будь-кого, хто отримує доступ до локальної файлової системи клієнта. Навіть якщо облікові дані є зашифрованими, зловмисник все-таки може просто відтворити файл cookie і, отже, увійти в систему як користувач, фактично не знаючи її облікових даних [16].

Багато додатків використовують HTTP для неавтентифікованих частин програми та переключаються на HTTPS у точці входу. Якщо це так, то правильне місце для переходу на HTTPS - це коли сторінка входу завантажується в браузері, що дозволяє користувачеві перевіряти, чи сторінка справжня, перед введенням облікових даних. Проте звичайно зустрічаються програми, які завантажують саму сторінку входу за допомогою HTTP, а потім переключаються на HTTPS у точці подання облікових даних. Це небезпечно, оскільки користувач не може перевірити автентичність сторінки входу, і тому не має гарантії, що облікові дані буде надійно передаватися. Правильно розташований атакуючий може перехопити та змінити сторінку входу, змінюючи цільову URL-адресу форми входу для використання HTTP. На той час, коли проникливий користувач усвідомлює, що облікові дані були відправлені за допомогою HTTP, вони будуть скомпрометовані [17].

1.1.2.5 Функціональність зміни пароля

Дивно, що багато веб-програм не дають користувачам можливості змінювати свій пароль. Однак ця функціональність необхідна для добре продуманого механізму автентифікації з двох причин.

По-перше, періодичне внесення змін до пароля пом'якшує загрозу застосування пароля. Це зменшує вікно, в якому певний пароль може бути націлений на атаку вгадування. Це також зменшує вікно, в якому компрометований пароль може використовуватися без виявлення зловмисника [18]. По-друге, користувачі, які підозрюють, що їх паролі можуть бути скомпрометовані, повинні мати можливість

швидко змінювати свій пароль, щоб зменшити загрозу несанкціонованого використання.

Незважаючи на те, що це необхідна частина ефективного механізму автентифікації, функціональність зміни паролів часто є вразливою за дизайном. Уразливості, які навмисно уникають у основній функції входу, часто з'являються в функції зміни паролів. Багато функцій зміни пароля веб-програм доступні без автентифікації та виконують такі дії:

- надати докладне повідомлення про помилку, яке вказує на те, чи дійсна назва запитуваного користувача є дійсною [19];
- дозволити необмежені здогадки поля "існуючий пароль";
- перевірити, чи поля "Новий пароль" та "Підтвердити новий пароль" мають однакове значення лише після перевірки існуючого пароля, тим самим дозволяючи атаці успішно відкрити існуючий пароль [20].

Типова функція зміни паролю включає відносно велике логічне дерево рішень. Програмі необхідно ідентифікувати користувача, перевірити наявний існуючий пароль, інтегрувати його з будь-якими захисниками блокування облікових записів, порівнювати додані нові паролі один з одним і з правилами щодо якості паролів, а також відкоригувати будь-які умови помилки користувачеві належним чином. Завдяки цьому функції зміни паролів часто містять витончені логічні недоліки, які можуть бути використані для того, щоб зруйнувати весь механізм.

1.1.2.6 Функція забутого пароля

Як і функціональність зміни пароля, механізми для відновлення забутих паролів часто спричиняють проблеми, які, можливо, були вирішені в основній частині входу користувача.

Окрім цього діапазону дефектів, дизайнерські слабкі місця в функції забутого паролю часто роблять це найслабшим ланкою системи, за допомогою яких можна атакувати загальну логіку автентифікації програми. Часто можна знайти кілька видів слабких місць дизайну [21].

Функція забутого паролю часто передбачає подання користувачеві вторинного завдання замість основного входу, як показано на малюнку 1.4. Цей зразок часто набагато простіший для зловмисника для відповіді, ніж спроби вгадати пароль користувача. Питання про дівоче ім’я матері, важливі дати з життя, улюблені кольори тощо, як правило, матимуть набагато менший набір потенційних відповідей, ніж набір можливих паролів. Крім того, вони часто стосуються відомостей, які є загальнодоступними або визначеними зловмисниками можуть виявити з незначним ступенем зусиль.

Рисунок 1.4 – Приклад завдання для відновлення паролю

У багатьох випадках додаток дозволяє користувачам встановлювати власне запитання та відповідь на відновлення пароля під час реєстрації. Користувачі схильні створювати надзвичайно небезпечні або занадто прості запитання, мабуть, на фальшивому припущенні, що тільки вони можуть знати відповідь до створених запитань. Приклад: "Чи є у мене човен?" У цій ситуації зловмисник, який хоче отримати доступ, може використовувати автоматичну атаку для ітерації за допомогою списку перелічених або загальних імен користувачів, реєструвати всі виклики для відновлення пароля та вибрати ті, відповіді до яких є найпростішими чи інтуїтивно-зрозумілими [22].

Як і в разі зміни функцій пароля, розробники додатків зазвичай не помічають можливості грубого нав'язування відповіді на запит на відновлення пароля, навіть якщо вони блокують цю атаку на головній сторінці входу. Якщо додаток дозволяє без обмежень спробувати відповісти на запити на виправлення пароля, цілком ймовірно, що він буде скомпрометований визначеним злочинцем.

У деяких додатках завдання відновлення замінюється простою «підказкою» паролю, який налаштовується користувачами під час реєстрації. Користувачі зазвичай встановлюють надзвичайно очевидні підказки, можливо, навіть ті, які ідентичні самому паролю, на фальшивому припущенні, що тільки вони коли-небудь побачать їх. Знову ж таки, зловмисник зі списком загальних чи перелічених імен користувачів може легко захопити велику кількість підказок для паролів, а потім почати здогадуватися [23].

Механізм, за допомогою якого програма дозволяє користувачам відновити контроль над своїм обліковим записом після правильного реагування на виклик, часто є вразливими. Одним із достатньо надійних засобів здійснення цього є надсилання унікальної, незаперечної URL-адреси для відновлення, обмежена за часом на адресу електронної пошти, яку користувач надавав під час реєстрації. Відвідування цієї URL-адреси протягом кількох хвилин дає користувачеві можливість встановити новий пароль. Тим не менше, часто зустрічаються інші механізми для відновлення облікового запису, які небезпечні за дизайном.

Наприклад, деякі програми розкривають існуючий, забутий пароль користувачеві після успішного завершення виклику, дозволяючи зловмисникові користуватися обліковим записом назавжди, без ризику виявлення власником. Навіть якщо власник облікового запису згодом змінює випалений пароль, зловмисник може просто повторити ту ж проблему, щоб отримати новий пароль [24].

Деякі програми негайно перенаправляють користувача в автентифікований сеанс після успішного завершення процесу відновлення паролю, знову дозволяючи зловмисникові користуватися обліковим записом необмежено без виявлення і навіть не знати пароль користувача.

Деякі програми використовують механізм надсилання унікальної URL-адреси для відновлення, але надсилають її на адресу електронної пошти, вказану користувачем, у той час, коли процес завершено. Це не забезпечує абсолютно жодної підвищеної безпеки для процесу відновлення, окрім можливого входу в електронну адресу, що використовується зловмисником.

Деякі програми дозволяють користувачам скинути значення свого пароля безпосередньо після успішного завершення виклику і не надсилати жодного повідомлення електронною поштою користувачеві. Це означає, що зловмисник облікового запису не буде помічений, поки власник не намагатиметься ввійти в систему знову. Це може навіть залишатися непоміченим, якщо власник припускає, що він, напевно, забув свій пароль і, таким чином, скидає його. Зловмисник, який просто бажає отримати доступ до додатка, може поставити під загрозу інший обліковий запис користувача протягом певного періоду часу, і тому зможе продовжувати використовувати додаток на невизначений термін [25].

1.1.2.7 Функціональність "Запам’ятати мене"

Програми часто реалізують функції "Запам'ятати мене" як зручність для користувачів. Таким чином, користувачам не потрібно повторно вводити своє ім'я користувача та пароль кожного разу, коли вони використовують програму з певного комп'ютера. Ці функції часто є небезпечними за дизайном та роблять користувачів незахищеними.

Деякі функції "Запам'ятати мене" виконуються за допомогою простого cookie файлу. Коли цей cookie файл надається на початкову сторінку програми, програма використовує cookie файл для автентифікації користувача та створює сеанс програми для цієї людини, минаючи реєстраційну інформацію. Зловмисник може використовувати список загальних або перелічених імен користувача, щоб отримати повний доступ до програми без будь-якої автентифікації [26].

Деякі функції “Запам'ятати мене” встановлюють cookie файли, що не містять імені користувача, а своєрідний стійкий ідентифікатор сеансу, наприклад “RememberUser=1328”. Коли ідентифікатор надсилається на сторінку входу, програма шукає користувача, пов'язаного з ним, і створює сеанс програми для цього користувача. Як і звичайні токенси сесії, якщо ідентифікатори сеансу інших користувачів можуть бути передбачені або екстрапольовані, зловмисник може перебрати велику кількість потенційних ідентифікаторів, щоб знайти пов'язані з

користувачами додатків, а отже, отримати доступ до своїх облікових записів без перевірки автентичності.

Навіть якщо інформація, що зберігається для повторного ідентифікації користувачів, є відповідною захищеною (зашифрованою), щоб інші користувачі не могли її визначити або вгадати, ця інформація може бути вразливою для захоплення через помилку, наприклад, міжсторінкове скриптування, або зловмисником, який має локальний доступ до комп'ютера користувача [27].

1.1.2.8 Неунікальні імена користувачів

Деякі програми, які підтримують автоматичну реєстрацію, дозволяють користувачам вказувати своє власне ім'я користувача та не вимагають, щоб імена користувачів були унікальними. Хоча це трапляється рідко, деякі відомі програмні рішення мати такі проблеми.

Це є недоліком дизайну з двох причин.

По-перше, один користувач, який має ім'я користувача з іншим користувачем, також може вибрати той самий пароль, що і цей користувач, під час реєстрації або після зміни пароля. У такому разі програма відхиляє вибраний пароль другого користувача або дозволяє двом обліковим записам мати ідентичні облікові дані. У першу чергу, поведінка програми ефективно розкриває одному користувачеві облікові дані іншого користувача. У другому випадку наступні логіни одного користувача призводять до доступу до іншого облікового запису користувача.

По-друге, зловмисник може використовувати цю поведінку для успішного нападу, навіть якщо це може бути неможливо в інших місцях через обмеження на спроби входу в систему. Зловмисник може декілька разів зареєструвати певне ім'я користувача за допомогою різних паролів під час моніторингу для диференціальної відповіді, що вказує на те, що обліковий запис з таким ім'ям і паролем вже існує. Зловмисник перевірить пароль цільового користувача, не зробивши жодної спроби ввійти як цей користувач [28].

Неправильно розроблена функціональність автоматичної реєстрації також може забезпечити засіб для перебору імен користувачів. Якщо програма забороняє дублікати імен користувачів, зловмисник може намагатися зареєструвати велику кількість загальних імен користувачів, щоб ідентифікувати існуючі імена користувачів, які відхилені.

1.1.2.9 Прогнозовані імена користувачів

Деякі програми автоматично генерують ім'я користувача облікового запису відповідно до передбачуваної послідовності (наприклад, cust5331, cust5332 тощо). Коли програма веде себе так, атакуючий, який може розпізнати послідовність, може швидко перейти до потенційно вичерпного списку всіх дійсних імен користувачів, які можуть бути використані в якості основи для подальших атак. На відміну від методів переліку, які покладаються на повторювані запити, які керуються списками слів, це означає визначення імен користувачів можна здійснювати без утилізації з мінімальним взаємодією з додатком [29].

1.1.2.10 Передбачувані початкові паролі

У деяких програмах користувачі створюються одночасно або у великих партіях, і автоматично визначаються початкові паролі, які потім розповсюджуються. Засіб створення паролів може дозволити зловмисникові передбачити паролі інших користувачів програми. Така уразливість частішає в корпоративних додатках на основі внутрішніх мереж, наприклад, де кожен працівник має власний обліковий запис від свого імені та отримує друковане повідомлення про свій пароль.

У найбільш уразливих випадках всі користувачі отримують один і той самий пароль, або той, який тісно пов'язаний з їхнім ім'ям користувача чи функцією роботи. В інших випадках, сформовані паролі можуть містити послідовності, які можна ідентифікувати або здогадуватися при доступі до дуже малих зразків початкових паролів [30].

1.1.3 Вирішення проблем механізмів автентифікації

Ми розглянути найважливіші причини, що можуть спричинити проблеми автентифікації та безпеки системи загалом. Зараз ми розглянемо способи захисту від ймовірних вище згаданих проблем.

Реалізація рішення для безпечної автентифікації включає в себе спробу одночасного виконання кількох основних завдань безпеки та в багатьох випадках компенсувати інші цілі, такі як функціональність, зручність використання та загальні витрати. У деяких випадках "більша" безпека фактично може бути контр-продуктивною. Наприклад, змушуючи користувачів встановлювати дуже довгі паролі та часто змінювати їх, часто користувачі запитують свої паролі [31].

Через велику різноманітність можливих проблем для автентифікації та потенційно складних засобів захисту, які, можливо, доведеться застосувати до застосування для пом'якшення всіх їх, багато розробників та розробників додатків вирішують приймати певні загрози як дано, та концентруватися на запобіганні найбільш серйозних нападів. Ось кілька факторів, які слід враховувати, виділяючи відповідний баланс:

- критичність безпеки з урахуванням функціональних можливостей, які пропонує програма;
- ступінь терпіння користувачів та роботи з різними типами контроль автентифікації;
- вартість підтримки менш зручної системи;
- фінансова вартість конкуруючих альтернатив стосовно доходу, який може бути сформований за заявкою, або вартістю активів, які вона захищає.

1.1.3.1 Використання сильних даних для автентифікації

Необхідно забезпечити відповідні мінімальні вимоги щодо якості пароля. Вони можуть включати правила щодо мінімальної довжини; поява алфавітних, цифрових

та друкарських символів; поява як великих, так і нижніх регістрів; уникнення словникових слів, імен та інших загальних паролів; запобігання встановленню пароля на ім'я користувача; і запобігання подібності або збігу з раніше встановленими паролями. Як і більшість заходів безпеки, різні вимоги щодо якості паролів можуть бути доречними для різних категорій користувачів [32].

Назви користувачів повинні бути унікальними.

Будь-які імена користувачів і паролі, створені системою, повинні створюватися з достатньою ентропією, яку неможливо домогтися при послідовності чи прогнозуванні - навіть зловмисник, який отримує доступ до великої вибірки послідовно створюваних випадків.

Користувачам слід дозволити встановлювати достатньо сильні паролі. Наприклад, допускаються довгі паролі та широкий діапазон символів.

1.1.3.2 Керування автентифікаційними даними приховано

Всі облікові дані повинні бути створені, збережені та передані таким чином, що не призводить до несанкціонованого розкриття інформації.

Усі комунікації з клієнтом-сервером повинні бути захищені за допомогою добре встановленої криптографічної технології, такої як SSL. Користувальницькі рішення для захисту даних під час транзиту не є необхідними або бажаними [33].

Якщо вважатиметься кращим використовувати HTTP для неавтентифікованих областей програми, переконайтеся, що сама форма реєстрації завантажується за допомогою HTTPS, а не перемикається на HTTPS у точці входу в систему.

Для передачі облікових даних на сервер слід використовувати тільки запити POST. Повноваження не можна розміщувати в параметрах URL-адреси або файлах cookie. Повноваження ніколи не повинні передаватися назад клієнту, навіть у параметрах для HTTP-перенаправлення.

Усі компоненти додатків на стороні сервера повинні зберігати облікові дані у спосіб, який не дозволяє легко відновити їх початкові значення, навіть якщо зловмисник отримує повний доступ до всіх відповідних даних в базі даних програми.

Звичайним засобом досягнення цієї мети є використання сильної хеш-функції (така як SHA-256), належним чином засолена, щоб зменшити ефективність попередньо розроблених оффлайн атак. Сіль має бути специфічною для облікового запису, у якого є пароль, таким чином, щоб зловмисник не міг відтворити чи замінити хеш-значення [34].

Клієнтська функція "Пам'ятайте мене" функціональність повинна, загалом, пам'ятати лише такі незбережені елементи, як імена користувачів. У менш критично важливих додатках може вважатися доцільним, щоб користувачі могли вмикати об'єкт для запам'ятовування паролів. У цій ситуації клієнту не слід зберігати чіткі текстові дані (пароль слід зберігати в оборотному порядку за допомогою ключа, відомий лише серверу). Крім того, користувачам слід попереджати про ризики зловмисника, який має фізичний доступ до свого комп'ютера або віддалений комп'ютер. Особливу увагу слід приділити усуненню проблем між сценаріями між додатками, які можуть використовуватися для крадіжки збережених облікових даних.

Необхідно реалізувати установку для зміни пароля, і користувачам доведеться періодично змінювати свій пароль.

Якщо облікові дані для нових облікових записів поширюються на користувачів поза межами діапазону, їх слід надсилати якомога безпечніше, і вони мають бути обмеженими у часі. Користувач повинен бути зобов'язаний змінити їх при першому вході, і йому слід сказати, що знищує зв'язок після першого використання.

Якщо потрібно, розгляньте можливість зафіксувати деякі вхідні дані користувача (наприклад, окремі літери з запам'ятовуваного слова) за допомогою спадних меню, а не текстових полів. Це дозволить запобігти встановленню будь-яких клавіатурних шпигунів на комп'ютері користувача з усіх даних, які надає користувач. (Зауважте, однак, що простий клавіатурний шпигун - це лише один із засобів, за допомогою якого зловмисник може захопити вхід користувача. Якщо він або вона вже скомпрометувала комп'ютер користувача, в принципі зловмисник може зареєструвати будь-який тип події, включаючи рухи миші, подавати форму над HTTPS та захопленням екрана.)

1.1.3.3 Коректна перевірка повноваження

Паролі повинні бути перевірені в повному обсязі, тобто в чутливих до регістру спосіб, без фільтрування або зміни будь-яких символів і без обрізання пароля [35].

Заявка повинна бути агресивною, захищаючи себе від непередбачених подій, що відбуваються під час обробки входу. Наприклад, залежно від мови розробки, що використовується, програма повинна використовувати загальні обробники виключень навколо всіх викликів API. Вони повинні явно видалити всі сеансові та методологічні локальні дані, які використовуються для керування станом обробки входу, і явно скасовує поточний сеанс, тим самим викликаючи вимушений вихід з сервера, навіть якщо автентифікація якимось чином обходить.

Усі логіки автентифікації повинні бути ретельно переглянуті як псевдокодування, так і як фактичний вихідний код програми, для виявлення логічних помилок, таких як умови безвідмовної роботи.

Якщо реалізовано функціональність для підтримки персоналізації користувачів, це необхідно суворо контролювати, щоб гарантувати, що його не можна зловживати для отримання неавторизованого доступу. Через критичність функціональності часто варто вилучити цю функцію з програми, що звертається до громадськості, і реалізовувати її лише для внутрішніх адміністративних користувачів, для чого слід використовувати жорсткий контроль та перевірку використання ідентифікації.

Необхідно суворо контролювати багаторівневі логіни, щоб перешкоджати втручанню переходи та відносини між етапами:

Усі дані про прогрес на етапах та результати попередніх завдань валідації повинні зберігатися на об'єкті сеансу на стороні сервера та ніколи не повинні передаватися чи відкриватися від клієнта [36].

Користувач не повинен подавати більше ніж один раз інформацію, і користувачеві не повинно бути ніяких засобів змінити дані, які вже були зібрані та / або перевірені. Там, де елемент даних, таких як ім'я користувача, використовується

на кількох етапах, це слід зберігати в змінній сесії під час першого збору та посилання на нього згодом.

Першим завданням, що виконується на кожному етапі, повинно бути перевірка того, що усі попередні етапи були правильно виконані. Якщо це не так, спробу автентифікації слід негайно позначити як погану.

Щоб уникнути витоку інформації про те, який етап входу в систему не вдалося (це дозволило б зловмисник націлювати на кожну стадію по черзі), додаток завжди повинен проходити через всі етапи входу до системи, навіть якщо користувач не зміг правильно виконати попередні етапи , і навіть якщо оригінальне ім'я користувача було недійсним. Пройшовши всі етапи, додаток повинне представляти загальне повідомлення "не вдалося ввійти" під час завершення етапу завершення, не надаючи ніякої інформації про те, де сталася помилка.

Коли процес входу включає випадкове змінення питання, переконайтесь, що зловмисник не може ефективно вибирати власне запитання:

Завжди використовуйте багатоетапний процес, в якому користувачі ідентифікують себе на початковому етапі, і випадкове зміна питання представляється їм на більш пізньому етапі [37].

Коли даному користувачеві було представлено певне різне питання, зберігайте це питання в своєму постійному користувацькому профілі та переконайтеся, що для кожного користувача спробує ввійти до одного і того ж користувача, доки він не буде успішно відповідати на нього.

Коли користувачеві представляється користувацьке запитання, зберігайте питання, яке було задано протягом сеансу на стороні сервера, а не приховане поле у формі HTML, і перевірте наступну відповідь на це збережене питання.

1.1.3.4 Запобігання атакам грубої сили

Необхідно дотримуватися заходів у всіх різних завданнях, що виконуються функцією автентифікації, для запобігання атакам, які намагаються вирішити ці

проблеми за допомогою автоматизації. Це включає в себе сам логін, а також функції зміни пароля, відновлення з ситуації забутого пароля тощо.

Використання непередбачуваних імен користувачів та запобігання їх переліку становить значну перешкоду для повністю сліпого атак грубої сили і вимагає, щоб зловмисник якимось чином виявив одне або декілька конкретних імен користувачів перед тим, як встановити атаку.

Деякі критично важливі програми (наприклад, онлайн-банки) просто блокують обліковий запис після невеликої кількості невдалих логінів (наприклад, трьох). Вони також вимагають, щоб власник облікового запису здійснював різні дії поза базою для повторної активації облікового запису, наприклад, зателефонувавши на підтримку клієнтів та відповідаючи на низку питань безпеки. Недоліки цієї політики полягають у тому, що це дозволяє зловмисникові відмовити у наданні послуги законним користувачам, повторно вимикаючи їхні облікові записи та витрати на надання послуги відновлення облікового запису. Більш збалансована політика, придатна для більшості додатків, захищених мережею, - це призупинення облікових записів на короткий період часу (наприклад, 30 хвилин) після невеликої кількості невдалих спроб входу (наприклад, трьох). Це служить для масового сповільнення будь-якої атаки на виявлення паролів, одночасно пом'якшуючи ризик атак на відмову в обслуговуванні, а також зменшити роботу центрів обробки викликів [38].

Якщо застосовується політика тимчасового призупинення дії облікового запису, то слід дотримуватися заходів щодо забезпечення його ефективності.

По-перше, щоб запобігти витоку інформації, що веде до переліку імен користувачів, програма ніколи не повинна вказувати на те, що будь-який конкретний обліковий запис призупинено. Скоріше, він повинен реагувати на будь-яку серію невдалих логінів, навіть тих, що використовують неправильне ім'я користувача, з повідомленням про те, що облікові записи призупиняються, якщо виникають декілька помилок, і що користувач повинен спробувати пізніше (як тільки обговорювалися).

Показники політики не повинні розкриватися для користувачів. Просто розповідаючи законним користувачам "Спробувати ще раз пізніше" не зменшує їх якість обслуговування. Проте інформування атакуючого про те, скільки невдалих

спроб переноситься, і як довго це період призупинення, дозволяє йому оптимізувати будь-які спроби продовжувати вгадати паролі, незважаючи на політику.

Якщо обліковий запис призупинено, спроби входу в систему слід відхилити, навіть не перевіряючи облікові дані. Деякі програми, які запровадили політику призупинення, залишаються вразливими до насильницьких дій, оскільки вони продовжують повну обробку спроб входу під час періоду призупинення, і вони подають тонко (чи не так тонко) інше повідомлення, коли подаються дійсні облікові дані. Така поведінка дає змогу ефективному нападу на базові сили діяти на повній швидкості незалежно від політики призупинення.

Заборонні контрзаходи, такі як блокування облікових записів, не допомагають захистити від одного типу атаки грубої сили, яка часто є високоефективною - повторюючи довгий список перелічених імен користувачів, перевіряючи один слабкий пароль, наприклад пароль. Наприклад, якщо п'ять невдалих спроб викликати призупинення облікового запису, це означає, що зловмисник може спробувати чотири різні паролі на кожному обліковому записі, не створюючи жодних порушень для користувачів. У типовому додатку, що містить багато слабких паролів, такий зловмисник може скомпрометувати багато облікових записів.

Ефективність цього виду атаки, звичайно, буде суттєво зменшена, якщо інші сфери механізму автентифікації розроблені надійно. Якщо імена користувачів не можуть бути перераховані або надійно передбачені, зловмисник сповільниться через необхідність виконувати втручання з грубої сили, вгадуючи імена користувачів. І якщо існують сильні вимоги щодо якості пароля, набагато менш імовірно, що зловмисник вибере пароль для тестування, який навіть вибрав один користувач програми [39].

На додаток до цих елементів керування програма може спеціально захистити себе від такої атаки, використовуючи CAPTCHA (англ. "Completely Automated Public Turing test to tell Computers and Humans Apart") на кожній сторінці, яка може стати об'єктом насильницької боротьби атаки.

Приклад CAPTCHA зображений на рисунку 1.5.

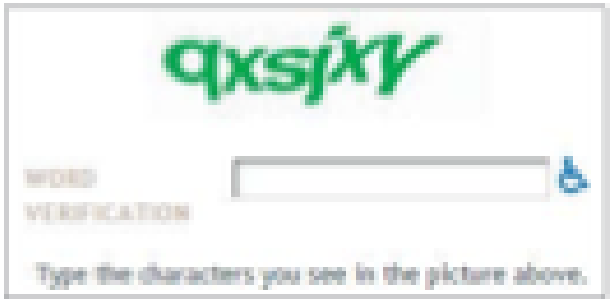

Рисунок 1.5 – Приклад CAPTCHA

Якщо це ефективно, цей захід може запобігти будь-якому автоматичному поданню даних на будь-яку сторінку додатка, тим самим зберігаючи всі можливі атаки з паролем, що виконуються вручну. Зауважте, що багато досліджень було зроблено за технологіями CAPTCHA, і в деяких випадках автоматичні напади на них були надійними. Крім того, деякі зловмисники, як відомо, розробляють змагання з вирішення CAPTCHA, в яких нездоланні члени громадськості використовуються як безпілотники для надання допомоги атакуючому. Проте, навіть якщо певний виклик не є цілком ефективним, він все одно призведе до того, що більшість випадкових хакерів захоплюються та виявляють застосування, яка не використовує цю техніку.

1.1.3.5 Запобігання неправильному використанню функції зміни пароля

Функція зміни паролю завжди повинна бути реалізована, щоб періодично закінчився термін дії пароля (якщо потрібно), а користувачам дозволяється змінювати паролі, якщо вони хочуть з будь-якої причини. Як ключовий механізм забезпечення безпеки, це необхідно добре захистити від нецільового використання.

Функція повинна бути доступною лише з автентифікованого сеансу.

Не повинно бути ніякого способу надати ім'я користувача явно або через приховане поле форми або файли cookie. Користувачі не мають законної потреби спробувати змінювати паролі інших людей.

Як поглиблене вимірювання захисту, ця функція повинна бути захищена від несанкціонованого доступу, отриманого через деякі інші дефекти безпеки в додатку,

такі як вразливість підхожого до виявлення сеансу, міжсистемний сценаріїв або навіть неавтоматичний термінал. З цією метою користувачам потрібно буде повторно ввести свій існуючий пароль.

Новий пароль потрібно вводити двічі, щоб запобігти помилкам. Програма має порівняти поля "новий пароль" та "підтвердити новий пароль" як перший крок і повернути інформаційну помилку, якщо вони не збігаються.

Функція повинна запобігти різним атакам, які можуть бути зроблені проти основного механізму входу. Необхідно використовувати єдине загальне повідомлення про помилку, щоб повідомити користувачів про будь-які помилки в існуючих облікових записах, і ця функція повинна бути тимчасово призупинена після невеликої кількості невдалих спроб зміни пароля.

Користувачам слід повідомляти нестандартні (наприклад, по електронній пошті), що їхній пароль був змінений, але повідомлення не повинно містити ні своїх старих, ні нових облікових даних.

1.1.3.6 Запобігання неправильному використанню функції відновлення облікового запису

У найбільш критично важливих додатках, таких як онлайн-банкінг, відновлення облікового запису у випадку забутого пароля здійснюється поза межами діапазону. Користувач повинен здійснити телефонний дзвінок і відповісти на ряд питань безпеки, а нові облікові дані або код активації також надсилаються поза межами діапазону (за допомогою звичайної пошти) на зареєстровану домашню адресу користувача. Більшість програм не хочуть або потребують такого рівня безпеки, тому функція автоматичного відновлення може бути доречною.

Добре розроблений механізм відновлення паролю повинен запобігати зловживання сторонами облікових записів сторонніми особами та мінімізації будь-яких порушень законних користувачів.

Функції, такі як "підказки" для паролів, ніколи не повинні використовуватися, оскільки вони в основному допомагають атакуючому траліку для облікових записів, які мають очевидні підказки.

Найкраще автоматизоване рішення, яке дає змогу користувачам відновити контроль над обліковими записами, - надіслати користувачеві електронну пошту унікальну, обмежену за часом, неперевершену, одноразову URL-адресу для відновлення. Це повідомлення електронної пошти слід надіслати на адресу, яку користувач надав під час реєстрації. Відвідування URL-адреси дозволяє користувачеві встановити новий пароль. Після цього потрібно надіслати друге повідомлення електронної пошти, яке вказує на зміну пароля. Щоб запобігти зловмиснику відмовляти в наданні послуг користувачам, постійно запитуючи електронну пошту для відновлення пароля, існуючі облікові дані користувача повинні залишатися чинними, доки вони не будуть змінені [40].

Щоб додатково захистити від несанкціонованого доступу, додатки можуть надавати користувачам додаткову проблему, яку вони повинні виконати, перш ніж отримати доступ до функції скидання пароля. Переконайтеся, що дизайн цієї проблеми не вносить нових вразливостей:

Виклик має реалізовувати одне і те ж питання або набір питань для кожного, що передбачені заявкою під час реєстрації. Якщо користувачі надають свої власні виклики, імовірно, що деякі з них будуть слабкими, і це також дасть змогу зловмисникові перерахувати дійсні облікові записи, визначаючи ті, у кого є задача.

Відповіді на виклик повинні містити достатню ентропію, яку їх легко угадати. Наприклад, запитуючи користувача про ім'я його першої школи, краще запитувати його улюблений колір.

Облікові записи повинні бути тимчасово призупинені після ряду невдалих спроб завершити виклик, щоб запобігти атакам грубої сили.

Заявка не повинна викидати будь-яку інформацію у разі невдалої відповіді на виклик - щодо дійсності імені користувача, призупинення дії облікового запису тощо.

Для успішного завершення завдання слід описати раніше описаний процес, в якому повідомлення надсилається на зареєстровану адресу користувача, що містить

URL-адресу для повторної активації. За жодних обставин програма не повинна розкривати забутий пароль користувача або просто залишити користувача в автентифікований сеанс. Навіть перехід безпосередньо до функції скидання пароля небажаний. Відповідь на запит на відновлення облікового запису, загалом, буде простішим для зловмисника, щоб вгадати, ніж вихідний пароль, тому не слід покладатися на це самостійно, щоб автентифікувати користувача.

1.1.3.7 Журнал, монітор та сповіщення

Програма повинна реєструвати всі пов'язані з автентифікацією події, включаючи вхід до системи, вихід з системи, зміну пароля, скидання пароля, призупинення облікового запису та відновлення облікового запису. У відповідних випадках, як невдалі, так і успішні спроби повинні бути зареєстровані. Журнали повинні містити всі релевантні дані (наприклад, ім'я користувача та IP-адресу), але немає секретності (наприклад, паролів). Журнали повинні бути сильно захищені від несанкціонованого доступу, оскільки вони є критичним джерелом витоку інформації.

Аномалії у випадках автентифікації повинні оброблятися програмним забезпеченням у режимі реального часу, що попереджають про це та запобігають втручанню. Наприклад, адміністратори аплікацій повинні бути інформовані про шаблони, що вказують на атаки грубої сили, з тим щоб можна було враховувати відповідні захисні та наступальні заходи.

Користувачам слід повідомляти про непридатність будь-яких критичних подій безпеки. Наприклад, програма повинна надіслати повідомлення на зареєстровану електронну адресу користувача, коли він змінює свій пароль.

Користувачам слід повідомляти про появу часто зустрічаються подій безпеки. Наприклад, після успішного входу в систему програма повинна інформувати користувачів про час і джерело IP / домен останнього входу та кількості недійсних спроб входу в систему, зроблених з того часу. Якщо користувачеві повідомляється, що її обліковий запис піддається атаці на вгадування паролем, вона частіше змінює пароль і встановлює його як сильне значення.

Функції автентифікації - це, мабуть, найвизначніша ціль у типовій частині атаки програми. За визначенням, вони можуть бути досягнуті непривілейованими, анонімними користувачами. Якщо вони зламані, вони надають доступ до захищених функцій та конфіденційних даних. Вони лежать в основі механізмів безпеки, які програма використовує для захисту себе і є фронтом захисту від несанкціонованого доступу.

Механізми автентифікації в реальному світі містять безліч недоліків дизайну та реалізації. Ефективний штурм проти них повинен відбуватися систематично, використовуючи структуровану методологію, щоб працювати через всілякі можливості атаки. У багатьох випадках з'являються відкриті цілі - погані паролі, способи з'ясування імен користувачів, вразливість до нападів грубої сили. На іншому кінці спектру дефекти можуть бути дуже важко розкрити. Вони можуть вимагати ретельного аналізу складного процесу входу, щоб визначити припущення, яке було зроблено, і допомогти вам виявити тонкий логічний дефект, який може бути використаний для проходження прямо через двері [41].

Найважливіший урок під час нападу на функціональність автентифікації - це шукати скрізь. На додаток до основної форми входу можуть існувати функції для реєстрації нових облікових записів, зміни паролів, запам'ятовування паролів, відновлення забутих паролів та висування себе за інших користувачів. Кожна з них являє собою багату ціль потенційних дефектів, і проблеми, які були свідомо усунені в рамках однієї функції, часто виникають в інших. Інвестуйте час, щоб перевірити і перевірити кожен дюйм поверхні атаки, який ви можете знайти.

1.2 Аналіз проблем авторизації

Авторизація - це функція визначення прав доступу / привілеїв ресурсів, пов'язаних з інформаційною безпекою та комп'ютерною безпекою в цілому та доступом до контролю зокрема. [1] Більш офіційно "авторизація" - це визначення політики доступу. Наприклад, працівники людських ресурсів, як правило, мають право на доступ до записів працівників, і ця політика зазвичай оформляється як правила

контролю доступу в комп'ютерній системі. Під час роботи система використовує правила контролю доступу, щоб вирішити, чи доступ до запитів від (аутентифікації) повинен бути схвалений (наданий) або відхилений (відхилений) [2]. Ресурси включають окремі файли або дані об'єкта, комп'ютерні програми, комп'ютерні пристрої та функціональні можливості, надані комп'ютерними програмами. Прикладами споживачів є комп'ютери, комп'ютерне програмне забезпечення та інше обладнання на комп'ютері.

Щоб зрозуміти проблеми авторизації, потрібно розглянути два важливі принципи, що описують рішення до основних проблем даного підрозділу.

1.1.4 Рольовий контроль доступу

У безпеці комп'ютерних систем, рольовий контроль доступу (RBAC) – це підхід до обмеження доступу системи до авторизованих користувачів. Він використовується більшістю підприємств з більш ніж 500 працівниками і може здійснювати обов'язковий контроль доступу (MAC) або дискреційний контроль доступу (DAC). RBAC іноді називають рольовою безпекою.

RBAC - це механізм контролю доступу, який визначає ролі та привілеї. Компоненти RBAC, такі як ролеві дозволи, роль користувача та рольові ролеві взаємини, полегшують виконання завдань користувача. Дослідження NIST показало, що RBAC вирішує багато потреб комерційних та державних організацій. RBAC може використовуватися для полегшення адміністрування безпеки у великих організаціях із сотнями користувачів та тисячами дозволів. Незважаючи на те, що RBAC відрізняється від схем керування доступом MAC та DAC, він може застосовувати ці правила без жодних ускладнень.

В межах організації ролі створюються для виконання різних функцій роботи. Права на виконання певних операцій призначаються для певних ролей. Членам або персоналу (або іншим користувачам системи) призначаються певні ролі, а завдяки цим ролевим завданням вони отримують дозволи, необхідні для виконання певних системних функцій. Оскільки користувачам не призначаються дозволи

безпосередньо, а лише набувають їх через свою роль (або ролі), управління окремими правами користувачів стає просто призначенням відповідних ролей на обліковий запис користувача; це спрощує звичайні операції, такі як додавання користувача або зміна департаменту користувача.

Для RBAC визначено три основні правила.

Рольові завдання: суб'єкт може здійснювати дозвіл лише у тому випадку, якщо суб'єкт обрано або призначено роль.

Роль авторизації: активна роль суб'єкта повинна бути дозволена для цього предмета. За правилом 1 вище це правило гарантує, що користувачі зможуть брати на себе лише ті ролі, для яких вони санкціоновані.

Авторизація дозволу: суб'єкт може виконувати дозвіл лише у випадку дозволу для активної ролі суб'єкта. За правилами 1 і 2 це правило гарантує, що користувачі можуть виконувати лише дозволи, для яких вони санкціоновані [42].

Також можуть бути застосовані додаткові обмеження, а ролі можна поєднати в ієрархії, де ролі вищого рівня підкоряються правам, що належать підролям.

1.1.5 Принцип мінімальних привілеїв

Принцип мінімальних привілеїв, або просто мінімальні привілеї, – це принцип організації доступу до ресурсів, коли в той чи інший рівень абстракції від обчислювального середовища, кожен модуль (такий, як процес, користувач або програма, які ми розглядаємо) повинні мати доступ до такої інформації і ресурсів, які мінімально необхідні для успішного виконання його робочої мети.

Принцип означає надання користувачеві облікового запису або обробки лише тих привілеїв, які необхідні для виконання передбачуваної функції. Наприклад, обліковий запис користувача з єдиною метою створення резервних копій не потребує встановлення програмного забезпечення: отже, він має право лише виконувати резервне копіювання та резервне копіювання. Будь-які інші привілеї, такі як встановлення нового програмного забезпечення, заблоковано. Цей принцип поширюється також на користувача персонального комп'ютера, який зазвичай

працює в звичайному обліковому записі користувача, і відкриває привілейований, захищений паролем обліковий запис (тобто супер-користувач) лише тоді, коли ситуація абсолютно вимагає цього.

Коли застосовується до користувачів, використовуються також умови найменшого доступу користувачів або найменш привілейована обліковий запис користувача (LUA), що посилається на концепцію, що всі облікові записи користувачів завжди повинні працювати з якомога меншими привілеями, а також запускати програми з небагатьма пільги, як це можливо.

Принцип найменших привілеїв широко визнається важливим дизайном у підвищенні захисту даних та функціональності від помилок та зловмисної поведінки.

Розглянемо принципи даного підходу.

Краща стабільність системи. Коли код обмежується сферою змін, які вона може зробити до системи, то простіше перевірити його можливі дії та взаємодії з іншими програмами. На практиці, наприклад, програми, що працюють з обмеженими правами, не матимуть доступу для виконання операцій, які можуть призвести до збою машини або негативно впливати на інші програми, що працюють у тій самій системі. Краща система безпеки. Коли код обмежений загальносистемними діями, які він може виконувати, вразливості в одній програмі не можуть бути використані для експлуатації решти машини. Наприклад, корпорація Microsoft заявляє: "Запуск в стандартному користувацькому режимі надає клієнтам додатковий захист від випадкового пошкодження на рівні системи, спричиненого атаками" штовхання "та шкідливими програмами, такими як кореневі комплекти, шпигунські програми та невиявлені віруси".

Простота розгортання. Загалом, чим менше привілеїв для роботи програми, тим легше його розгортати в більш широкому середовищі. Як правило, це виникає з перших двох переваг: програми, які встановлюють драйвери пристроїв або вимагають підвищених привілеїв безпеки, зазвичай мають додаткові етапи їхнього розгортання. Наприклад, у Windows рішення без жодних драйверів пристроїв можна запустити безпосередньо без установки, тоді як драйвери пристроїв повинні бути встановлені

окремо за допомогою служби інсталятора Windows, щоб надавати водіям підвищені права.

На практиці існує декілька конкуруючих визначень справжніх мінімальних привілеїв. Оскільки складність програми зростає за експоненціальною швидкістю, то й кількість потенційних проблем, що робить прогностичний підхід непрактичним. Приклади включають значення змінних, які він може обробляти, адреси, які воно буде потрібно, або точне час, за яким такі речі будуть потрібні. Системи можливостей об'єкта дозволяють, наприклад, відкладати надання одноразової привілеї до часу, коли воно буде використовуватися.

На даний момент найближчим практичним підходом є усунення привілеїв, які можна вручну оцінити як непотрібні. Отриманий набір прав переважно перевищує дійсний мінімум необхідних привілеїв для процесу.

Іншим обмеженням є деталізація контролю над тим, що операційне середовище має переваги для окремих процесів.

На практиці дуже рідко можна контролювати доступ процесу до пам'яті, час обробки, адреси пристрою вводу / виводу з точністю, необхідною для полегшення лише точного набору привілеїв, який вимагатиме процес.

1.3 Аналіз проблем шифрування даних

Шифрування – це процес перетворення даних, яке виконується у посимвольній послідовності з метою одержання шифрованого тексту.

Дешифрування, відповідно, - це процес отримання інформації з зашифрованих джерел.

Щоб зрозуміти яке шифрування та дешифрування використовувати, потрібно розглянути методи шифрування та принципи їх роботи. Це допоможе з’ясувати основні проблеми шифрування, та способи вирішення даних проблем.

Спеціалісти комп’ютерної безпеки виділяють 2 методи шифрування: симетричне та асиметричне. Далі ми розглянемо кожен з цих методів.

1.1.6 Симетричне шифрування

Симетричне шифрування є найстарішою і найвідомішою технікою. Секретний ключ, який може бути числом, словом або просто рядком випадкових літер, застосовується до тексту повідомлення, щоб змінити вміст певним чином. Це може бути настільки ж простим, як переміщення кожної літери кількома місцями в алфавіті. Поки обидва відправники та одержувачі знають таємний ключ, вони можуть шифрувати та дешифрувати всі повідомлення, які використовують цю клавішу (рисунок 1.6).

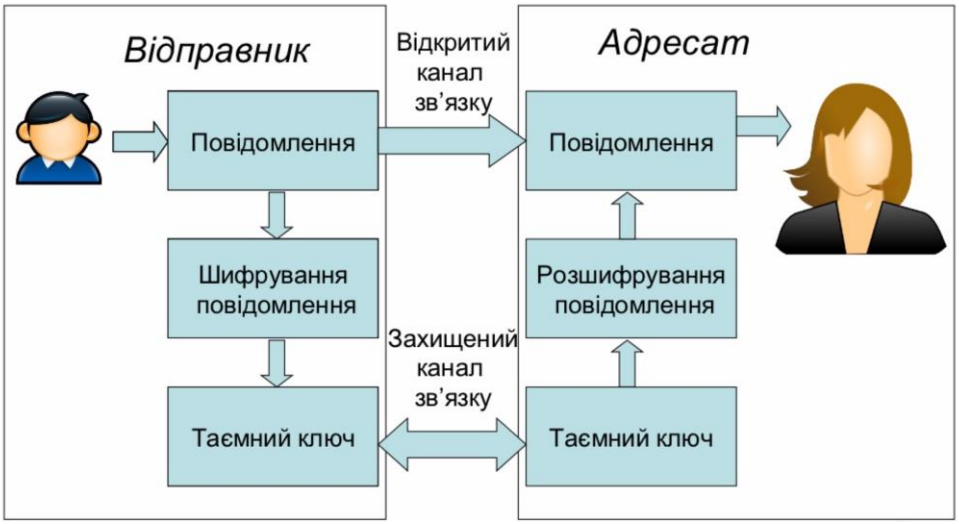

Рисунок 1.6 – Принцип роботи симетричної криптосистеми

Шифрування за допомогою симетричного ключа може використовувати як шифрування потоку, так і блокові шифри.

Потокове шифрування шифрують цифри (як правило, байтів) або букви (в шифрах заміни) повідомлення по одному за раз. Прикладом є Vigenere Cipher [43].

Шифрування блоків бере декілька бітів і шифрує їх як окремий блок, наповнюючи відкритий текст так, щоб він був кратний розміру блоку. Часто використовувались блоки з 64 бітами. Алгоритм Advanced Cryption Standard (AES), схвалений NIST в грудні 2001 року, і режим роботи блочного шифрування GCM використовують 128-бітні блоки.

Приклади популярних алгоритмів симетричного ключа: Twofish, Serpent, AES (Rijndael), Blowfish, CAST5, Kuznyechik, RC4, DES, 3DES.

1.1.7 Асиметричне шифрування

Криптографія з відкритим ключем або асиметрична криптографія - це будь-яка криптографічна система, яка використовує пару ключів: відкритий ключі, які можуть бути широко розповсюджені, та приватний ключ, відомі лише власнику. Це виконує дві функції: аутентифікація, де публічний ключ перевіряє, що власник парного приватного ключа надіслав повідомлення, і шифрування, де тільки парний приватний ключовий власник може розшифрувати повідомлення, зашифроване за допомогою відкритого ключа.

У системі шифрування відкритих ключів будь-яка людина може шифрувати повідомлення за допомогою відкритого ключа отримувача. Це зашифроване повідомлення може бути розшифровано лише з приватним ключем одержувача. Щоб бути практичним, створення публічного та приватного ключа-пари має бути обчислено економічно. Сила криптографічної системи відкритого ключа залежить від обчислювальних зусиль (робочого фактору в криптографії), необхідних для пошуку приватного ключа з його парного відкритого ключа. Ефективна безпека вимагає лише приватного ключа приватного характеру; відкритий ключ може бути відкрито розподілений без шкоди для безпеки.

Криптографічні системи з відкритим ключем часто покладаються на криптографічні алгоритми на основі математичних задач, які в даний час не допускають ефективного рішення, зокрема ті, що притаманні певній цілій факторизації, дискретному логарифмі та відносинам з еліптичними кривими. Алгоритми відкритого ключа, на відміну від симетричних алгоритмів ключів, не вимагають безпечного каналу для первинного обміну однією або декількома секретними ключами між сторонами [44].

Через обчислювальну складність асиметричного шифрування вона зазвичай використовується лише для невеликих блоків даних, як правило, для передачі

симетричного ключа шифрування (наприклад, ключ сеансу). Цей симетричний ключ потім використовується для шифрування решти потенційно довгої послідовності повідомлень. Симетричне шифрування / дешифрування базується на більш простих алгоритмах і набагато швидше.

У системі підписів відкритого ключа людина може поєднувати повідомлення з приватним ключем, щоб створити короткий цифровий підпис у повідомленні. Кожен, хто має відповідний відкритий ключ, може поєднувати повідомлення, передбачуваний цифровий підпис на ньому та відомий відкритий ключ, щоб перевірити, чи підпис був дійсним, тобто зробив власник відповідного приватного ключа. Зміна повідомлення, навіть заміна однієї літери, призведе до помилки підтвердження. У безпечній системі підписів це неможливо для обчислення для тих, хто не знає приватного ключа, щоб вивести його з відкритого ключа або будь-якої кількості підписів, або щоб знайти підпису на будь-якому повідомленні, для якого підпис ще не був помічений . Таким чином, автентичність повідомлення може бути продемонстрована підписом, якщо власник приватної ключа зберігає секретний пароль (рисунок 1.7).

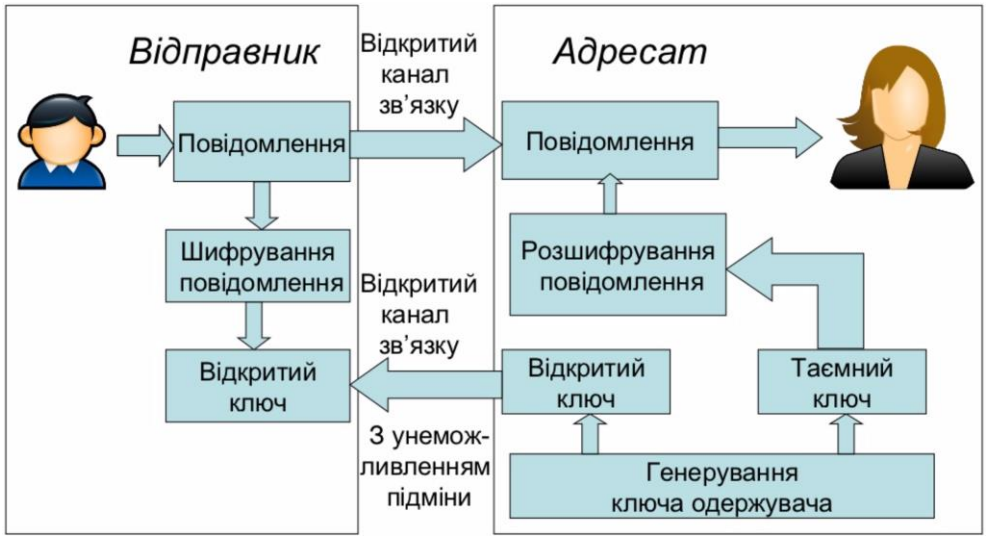

Рисунок 1.7 – Принцип роботи асиметричної криптосистеми

Алгоритми відкритого ключа є основними інгредієнтами безпеки в криптосистемах, додатках і протоколах. Вони підкріплюють різні стандарти Інтернету, такі як Transport Layer Security (TLS), S / MIME, PGP та GPG. Деякі

алгоритми публічного ключа забезпечують розповсюдження та секретність ключових слів (наприклад, обмін ключем Діффі-Хеллмана), деякі забезпечують цифрові підписи (наприклад, алгоритм цифрового підпису), а деякі - як (наприклад, RSA).

Криптографія відкритого ключа знаходить застосування, зокрема, в дисципліні інформаційної безпеки, інформаційної безпеки. Інформаційна безпека (IS) стосується всіх аспектів захисту електронних інформаційних активів від загроз безпеці [6]. Криптографія відкритого ключа використовується як спосіб забезпечення конфіденційності, автентичності та невідчутності електронних повідомлень та зберігання даних.

1.4 Аналіз проблем балансування навантаження

Перш ніж перейти до аналізу проблем балансування навантаження, потрібно розглянути механізми за яких здійснюється балансування: використання серверів для балансування, або застосування зворотних проксі-серверів.

1.1.8 Використання серверів для балансування

Балансування навантаження найчастіше застосовується, коли системі потрібно декілька серверів, тому що обсяг запитів занадто великий для ефективного оброблення лише одним сервером. Розгортання декількох серверів також усуває одну причину падіння системи, що робить її надійнішою. Найчастіше всі сервери містять один і той самий вміст, і робота балансувального механізму полягає в тому, щоб розподілити робоче навантаження таким чином, щоб максимально використовувати можливості кожного сервера, запобігати перевантаженню на будь-якому сервері та забезпечити якнайшвидшу реакцію для клієнта [45].

Балансування навантаження також може покращити роботу користувачів, зменшивши кількість відповідей з помилками, які бачить клієнт. Це відбувається, коли сервери виходять з ладу, і відводять свої запити до інших серверів і групі. У найпростішому виконанні балансувальник навантаження виявляє здоров'я сервера,

перехоплюючи відповіді з помилками. Перевірки здоров’я додатків є більш гнучким та складним способом, при якому балансування навантаження надсилає окремі запити на перевірку здоров'я і вимагає певного типу відповіді, щоб вважати сервіс здоровим.

Ще однією корисною функцією, яку забезпечують деякі механізми балансування навантаження, є однорідність сеансу, а це означає, що всі запити від конкретного клієнта надсилаються на той самий сервер. Незважаючи на те, що в теорії HTTP не має стану, у багатьох додатках потрібно зберігати інформацію про стан лише для того, щоб забезпечити їх основну функціональність (наприклад, кошик для покупок на сайті електронної комерції). Такі програми неефективні або навіть можуть зазнавати невдач у середовищі, де збалансоване навантаження, якщо балансувальник навантаження розподіляє запити в сеансі користувача на різні сервери, замість того, щоб спрямовувати їх на сервер, який відповідав на початковий запит.

1.1.9 Застосування зворотних проксі-серверів

Хоча розгортання механізму балансування навантаження має сенс тільки в тому випадку, якщо ви маєте кілька серверів, часто буває сенс розгорнути зворотний проксі-сервер навіть із одного веб-сервера або сервера додатків. Наприклад, проксі-сервер може виступати "публічним обличчям" певної системи. Його адреса є відкритою та доступною для зовнішнього світу. Перевагами цього підходу є підвищена безпека та покращена масштабованість та гнучкість.

Інформація про ваші внутрішні сервери не відображається і не розповсюджується поза вашою внутрішньою мережею, тому зловмисники не можуть отримати доступ до них безпосередньо для використання будь-яких вразливостей. Багато зворотних проксі-серверів містять функції, які допомагають захистити внутрішні сервери від атак розповсюджених відмов (-сервісів) (DDoS), наприклад, відхиляючи трафік від певних IP-адрес клієнта (чорні списки) або обмежуючи кількість з'єднань, прийнятих з кожного клієнта [46].

Оскільки клієнти бачать лише IP-адресу зворотного проксі-сервера, це дає можливість можете вільно змінювати конфігурацію інфраструктури прихованої серверної частини. Можна масштабувати кількість сервері: збільшувати та зменшувати їх кількість, щоб відповідати коливанням обсягу трафіку.

Іншою причиною для розгортання зворотного проксі-сервера є веб-прискорення - скорочення часу, необхідного для створення відповіді, і повернення його до клієнта. Техніки веб-прискорення включають наступні механізми:

- стиснення даних;
- припинення спілкування по SSL протоколу;
- кешування.

Стискання відповідей сервера перед поверненням їх до клієнта (наприклад, за допомогою gzip) зменшує необхідну пропускну здатність, що прискорює їх транзит через мережу.

Шифрування трафіку між клієнтами та серверами захищає його, оскільки він перетинає загальну мережу, таку як Інтернет. Але розшифрування та шифрування можуть бути обчислювальними витратами. Розшифровуючи вхідні запити та шифруючи вихідні відповіді сервера, зворотний проксі-сервер звільняє ресурси на серверах системи моделювання гідроакустичних процесів, які вони можуть потім присвятити своїй головній меті. Перед поверненням відповіді від сервера до клієнту, зворотний проксі-сервер зберігає його копію локально. Коли будь-який клієнт робить той самий запит, зворотний проксі-сервер може надати така саму відповідь з кеша замість пересилання запиту на внутрішній сервер основної системи моделювання. Це одночасно зменшує час відгуку клієнта та зменшує навантаження на основні сервери системи. Оскільки для користувача системи моделювання акустичних процесів спілкування з одним сервером є критичним, система захисту повинна бути побудована саме на сервері балансування, а не на зворотному проксі-сервері.

1.1.10 Алгоритми балансування навантаження

Різні алгоритми балансування навантаження забезпечують різні переваги.

Вибір методу балансування навантаження залежить від потреб основної системи.

- алгоритм Round Robin;
- алгоритм найменшої кількісті зв’язків;
- IP хешування.

При використанні алгоритму Round Robin запити розподіляються по групі серверів послідовно. Алгоритм найменшої кількісті зв’язків відправить новий запит на сервер з найменшими поточними з'єднаннями з клієнтами. Відносна обчислювальна потужність кожного сервера враховується для визначення того, який з них має найменший зв'язок. При IP-хешуванні, IP-адреса клієнта використовується для визначення того, який сервер отримує запит [47].

Round Robin балансування навантаження - це один з найпростіших способів розповсюдження запитів клієнтів на групі серверів. Переходячи до списку серверів у групі, Round Robin механізм по черзі пересилає запит клієнта на кожний сервер. Коли він досягає кінця списку, балансувальник повертається назад і починає спочатку (відправляє наступний запит на перший перерахований сервер, після цього на другий сервер і так далі).

Основна перевага балансування об'ємного навантаження полягає в тому, що це надзвичайно просте втілення. Однак це не завжди призводить до найбільш точного або ефективного розподілу трафіку, тому що багато балансувальників кругової завантажувальної навантаження вважають, що всі сервери однакові: наразі вони працюють з тим самим завантаженням і мають однакову обсяг пам'яті та обчислень. Наступні варіанти алгоритму круглого циклу враховують додаткові фактори та можуть призвести до більшого балансування навантаження:

Зважене Round Robin балансування – вага призначається для кожного сервера на основі критеріїв, обраних адміністратором сайту; найбільш часто використовуваний критерій - це спроможність сервера обробляти трафік. Чим вище вага, тим більша частка клієнтських запитів, які отримує сервер. Якщо, наприклад, для сервера A призначається вага 3, а сервер B - вага 1, балансувальник навантаження пересилає 3 запити до сервера A для кожної одного запиту, що надсилається до сервера B.

При динамічному Round Robin балансуванні, вага кожного динамічного сервера призначається на основі даних в режимі реального часу про поточну навантаження сервера та пропускну здатність.

1.5 Аналіз проблем стиснення даних

Стиснення є важливим способом підвищення продуктивності системи. Для деяких документів зменшення розміру до 70% знижує потребу пропускної спроможності. Протягом багатьох років алгоритми також стали більш ефективними, а нові механізми підтримуються клієнтами та серверами.

На практиці розробникам не потрібно реалізувати механізми стиснення, і веб-переглядачі та сервери вже реалізовані, але вони повинні бути впевнені, що сервер налаштований належним чином.

Стиснення відбувається на трьох різних рівнях. Спочатку деякі формати файлів стискуються з конкретними оптимізованими методами, потім загальне шифрування може відбутися на рівні HTTP (ресурс передається стиснутою від кінця до кінця), і, нарешті, стиснення можна визначити на рівні з'єднання між двома вузлами HTTP-з'єднання.

Кожен тип даних має певний надлишок, тобто “змарнований простір”. Якщо текст має 80% надлишку

Кожен тип даних має деяку надмірність, тобто “втрачений простір”. Якщо для тексту це значення зазвичай 60%, цей показник може бути набагато вищим для деяких інших засобів масової інформації, таких як аудіо та відео. На відміну від тексту, ці інші типи носіїв забирають багато місця для зберігання, і необхідність відновити цей втрачений простір з'явилася дуже рано. Інженери розробили оптимізований алгоритм стиснення, який використовується форматами файлів, призначеними для цієї конкретної мети. Алгоритми стиснення, що використовуються для файлів, можна згрупувати у дві широкі категорії.

Перша категорія називається loss-less стиснення, тобто стиснення без втрат, або з втратами, що не впливають на якість контенту, або є непомітними для кінцевого

користувача. Даний процес перевіряє байт за байтом порівнюючи з оригіналом. Для зображень gif або png використовують стиснення без втрат.

Друга категорія – lossy стиснення, тобто стиснення з незначними втратами, що можуть бути помітні під час аналізу двох версій, але мінімально помітні для звичайного користувача. Цикл змінює вихідні дані, непомітно для користувача. Більшість форматів відео в Інтернеті, та jpeg – яскраві приклади lossy стиснення.

Деякі формати можуть бути використані як для обох видів стискання, наприклад, webp, і, зазвичай, алгоритм стиснення може бути налаштований для стискання більшої або меншої кількості інформації, що, безумовно, призводить до меншої або більшої якості змісту даного файлу. Для кращої роботи системи ідеально підібрати якомога більше стиснення, при цьому зберігаючи прийнятний рівень якості. Для зображень, створених механізмом стиснення, може бути недостатньо оптимізованим для Інтернету; рекомендується використовувати інструменти, які максимально стискатимуть до потрібної якості. Є багато інструментів, які спеціалізуються на цьому.

Алгоритми lossy, як правило, більш ефективні, ніж алгоритми loss-less.

1.1.11 Стиснення end-to-end

Стиснення "від кінця до кінця" означає стиснення тіла повідомлення, яке відправляє сервер, і гарантування того, що це тіло буде незмінним до досягнення клієнта. Незалежно від проміжних вузлів, вони залишають тіло недоторканим.

Всі сучасні браузери та сервери підтримують дане стиснення, і єдине, що потрібно вести переговори, - це алгоритм стиснення для використання. Ці алгоритми оптимізовані для тексту. У 90-х роках технологія стиснення швидко розвивалася, і численні послідовні алгоритми були додані до набору можливих варіантів. В даний час доречні лише два: gzip (рисунок 1.8), найпоширеніший, і br - новий претендент.

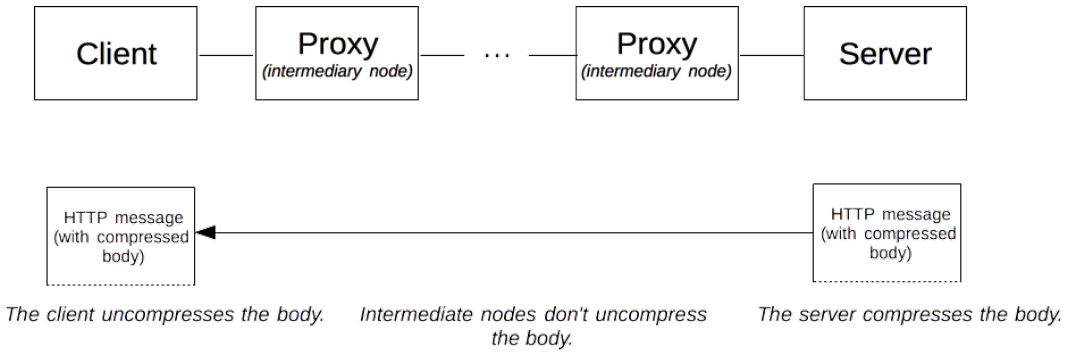

Рисунок 1.8 – Приклад end-to-end стиснення

Щоб вибрати алгоритм для використання, веб-переглядачі та сервери використовують узгодження вмісту. Браузер надсилає заголовок Accept-Encoding з підтримкою алгоритму, який він підтримує, і його порядок пріоритету, сервер вибирає його, використовує його для стиснення тіла відповіді та використовує заголовок Content-Encoding, щоб повідомити браузеру вибраний ним алгоритм. Оскільки узгодження вмісту було використано для вибору представлення на основі його кодування, сервер повинен відправити заголовок Vary, що містить принаймні Accept-Encoding разом з цим заголовком у відповідь; Таким чином, кеші зможуть кешувати різні представлення ресурсу (рисунок 1.9).

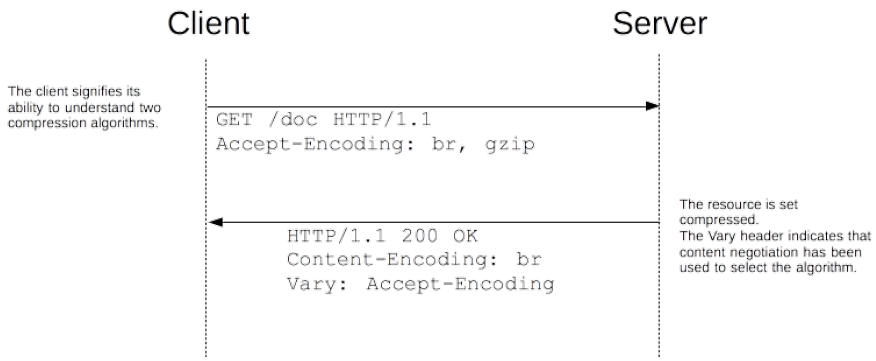

Рисунок 1.9 – Приклад використання Vary заголовку

1.1.12 Стиснення hop-by-hop

Стиснення hop-by-hop, хоча і схоже на стиснення end-to-end, відрізняється одним основним елементом: стиснення не відбувається на ресурсі на сервері, що

створює конкретне представлення, яке потім передає, а на тілі повідомлення між двома вузлами на шляху між клієнтом і сервером [48] (рисунок 1.10).

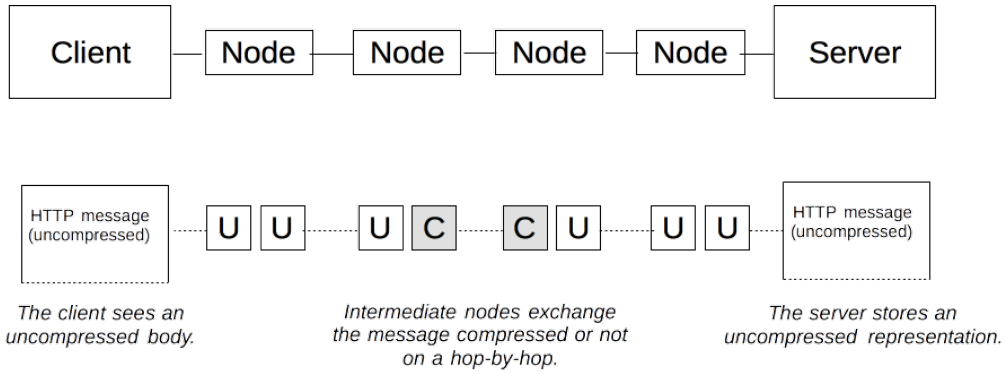

Рисунок 1.10 – Механізм роботи hop-by-hop стиснення

Для цього HTTP використовує механізм, подібний до узгодження вмісту для повного стиснення: вузол, який передає запит, декларує свою бажання, використовуючи заголовок TE, а інший вузол вибирає відповідний метод, застосовує його та вказує на його вибір заголовок Transfer-Encoding (рисунок 1.11).

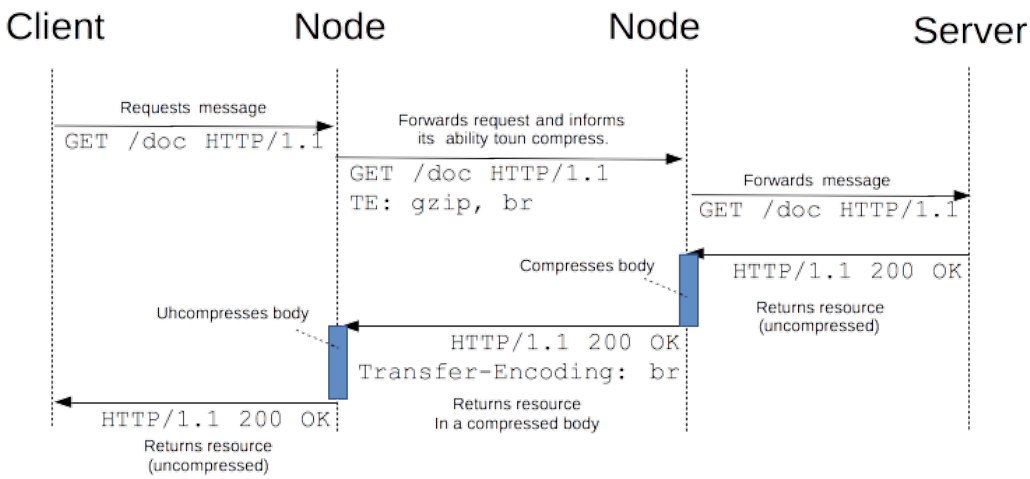

Рисунок 1.11 – Приклад hop-by-hop стиснення

На практиці, стиснення hop-by-hop є прозорим для сервера та клієнта, і його рідко використовують. TE та Transfer-Encoding використовуються переважно для відправки відповіді на шматки, що дозволяє розпочати передачу ресурсу, не знаючи його довжини.

Висновки до першого розділу

В даному розділі було детально досліджені та проаналізовані проблеми, що можуть виникнути в комп’ютерній безпеці системи на різних рівнях захисту. Даний розділ описує відомі та надійні механізми захисту та пояснює принципи їх роботи.

На основі першого розділу, можна уявити загальну картину системи безпеки для системи моделювання гідроакустичних процесів. Тут закладені принципи, що будуть відображені в архітектурі системи захисту, які будуть описані в наступному розділі.

Даний розділ дає вичерпний список технологій, які можуть бути використані для створення надійної системи безпеки. Кожна технологія згадана в даному розділі описана з урахуванням як позитивних так і негативних ефектів.

2. АРХІТЕКТУРНІ РІШЕННЯ ВИКОРИСТАНІ ДЛЯ СИСТЕМИ БЕЗПЕКИ

В попередньому розділі було розглянуто принципи захисту системи моделювання гідроакустичних процесів та технології, що можна використати при створенні системи захисту таких систем.

Детальний опис принципів та технологій захисту дає змогу зрозуміти, що саме потрібно для захисту системи гідроакустичних процесів, та якою повинна бути архітектура системи захисту.

В даному розділі, буде детально пояснена архітектура системи захисту та пояснено вибір тієї чи іншої технології, що була реалізована в даній системі. Спочатку, потрібно зрозуміти як система захисту буде інтегруватися з вже існуючою системою моделювання гідроакустичних процесів. Потім, потрібно розглянути архітектуру та технічний стек системи більш детально.

2.1 Інтегрування системи захисту до вже існуючої системи

Одною з головних цілей даної роботи є створення універсальної системи захисту, що зможе працювати з великою кількістю систем моделювання гідроакустичних процесів незалежно від їхньої архітектури.

Система захисту являє собою зворотній проксі-сервер написаний на мові програмування Java з використанням Spring Boot [49]. Як було пояснено в першому розділі, зворотній проксі-сервер часто застосовується для забезпечення принципів безпеки, оскільки він слугує посередником між своїми підопічними системами та рештою клієнтів в Інтернеті. Підопічні системи – це, свого роду, системи, що поєднанні локальною мережею, або об’єднанні для досягнення певної мети (наприклад, для моделювання гідроакустичних процесів).

В нашому випадку, будемо розглядати систему моделювання гідроакустичних процесів як єдиний HTTP сервер, що має велику кількість клієнтів, розташованих в різних куточках Інтернету. На рисунку 2.1 зображений приклад такої системи.

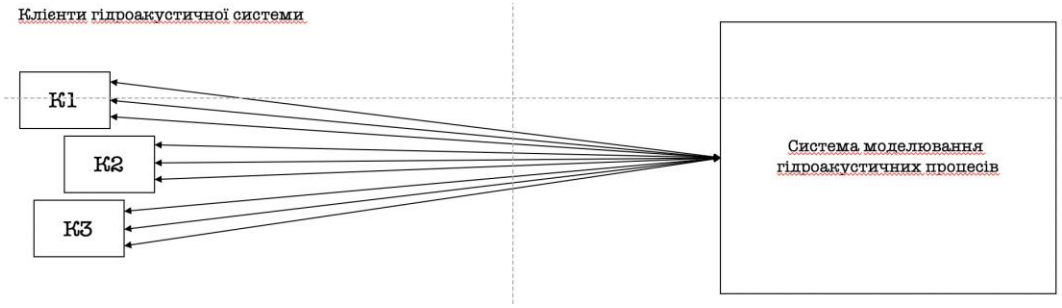

Рисунок 2.1 – Стандартна схема комунікації між клієнтами та системою

Оскільки система спілкується зі своїми клієнтами за допомогою певного мереживного протоколу, програмний продукт для захисту цієї системи має відповідати цьому протоколу спілкування. Дана система захисту також підтримує HTTPS протокол, тому на початковому етапі, ми можемо розглядати систему захисту одним з клієнтів системи моделювання.

Іншою важливою метою даного програмного рішення має бути проста на зручна інтеграція з системою моделювання гідроакустичних процесів.

Оскільки спілкування між клієнтами та сервером системи моделювання гідроакустичних процесів відбувається по HTTTS, ми можемо просто інтегрувати нашу систему та переправити всіх перевірених клієнтів на інтернет-адресу серверу захисту.

На рисунку 2.2 зображена система моделювання гідроакустичних процесів після її інтегрування з системою захисту.

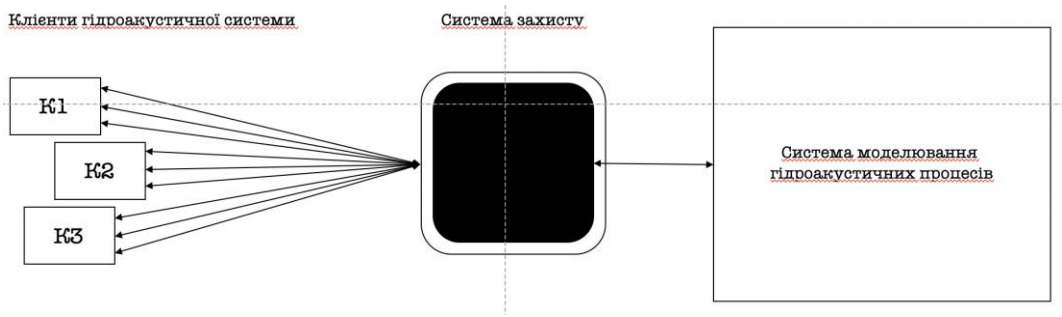

Рисунок 2.2 – Інтеграція системи безпеки

Після цього, клієнти системи моделювання гідроакустичних процесів насправді стануть клієнтами системи захисту. Але, оскільки, система захисту пропонує ідентичний інтерфейс доступу до потрібних операцій, з клієнтської частини неможливо визначати хто насправді буде обробляти її запити.

В кінці розділу буде розглянуто архітектуру API для налаштування системи захисту для системи моделювання гідроакустичних процесів.

2.2 Архітектура системи захисту

Архітектура сервера захисту можна умовно поділити на три складові:

- модуль контролерів;
- ядро системи;
- рівень бази даних.

Загальна архітектура системи зображена на рисунку 2.3.

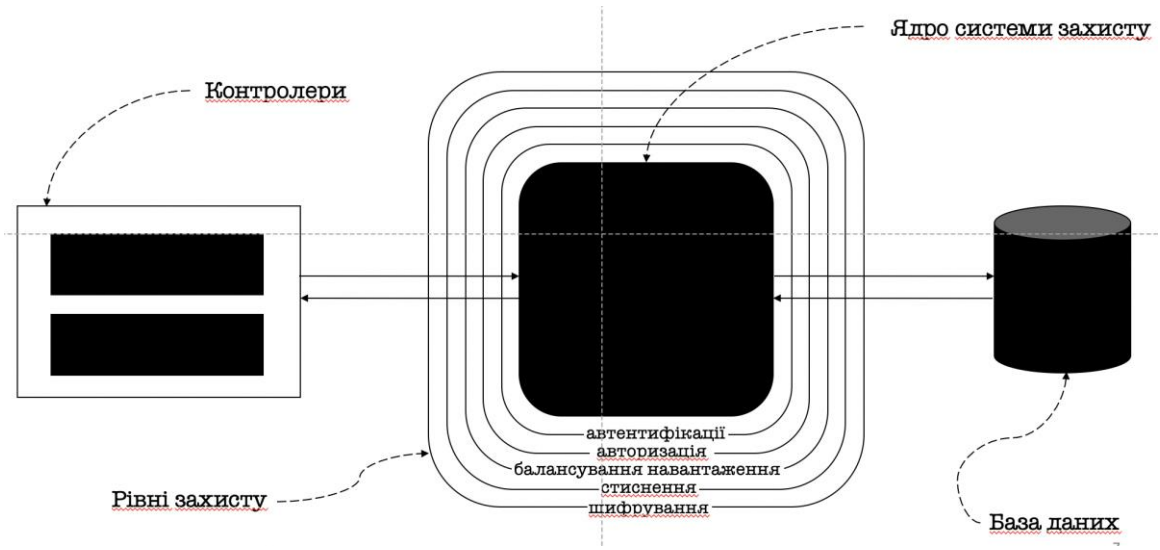

Рисунок 2.3 – Загальна архітектура системи захисту

Контролери відповідають за спілкування зі зовнішнім світом, тобто з будь-якими клієнтами, що мають доступ до даного серверу захисту. Частина контролерів складається з двох основних під-модулів: модель налаштування та модуль перенаправлення. Схематично, це зображено на рисунку 2.4.

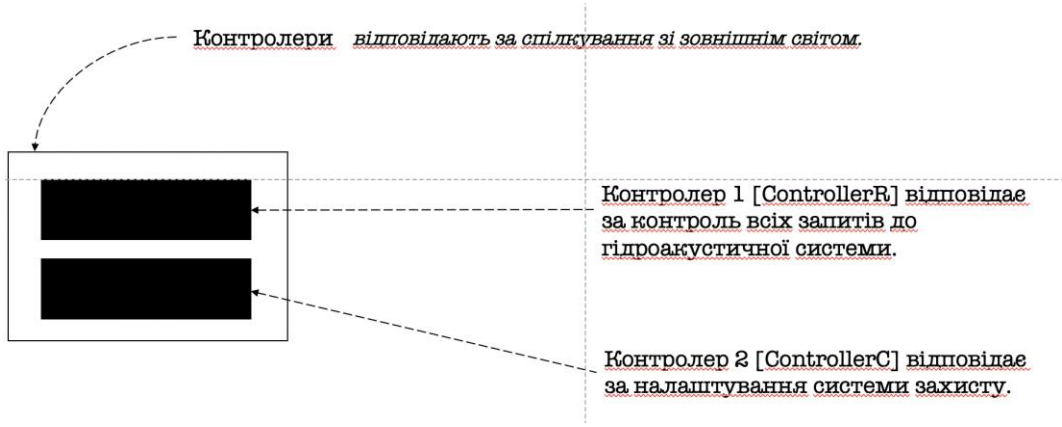

Рисунок 2.4 – Архітектура рівня контролерів

Під-модуль налаштування працює з API, що розроблений для налаштування конфігурацій безпеки на сервері для системи моделювання.

Під-модуль перенаправлення опрацьовує виклики, що стосуються системи моделювання гідроакустичних процесів. Спочатку, дана частина програми ідентифікує запит та перевіряє його контент, щоб впевнитися, що запит стосується системи моделювання.

Якщо запит був направлений до системи моделювання, інформація про запит копіюється та передається в ядро системи для подальшого опрацювання з застосуванням рівнів безпеки. Якщо запит стосується налаштування системи безпеки, він передається до під-модуля налаштування, де отримані дані валідуються.

Після валідування даних в під-модулі налаштування, коректні дані зберігаються в базі даних та застосовуються. При наступному запиті, нові налаштування безпеки вступають в силу.

Ядро системи відповідає за безпеку підопічної системи модулювання гідроакустичних процесів та опрацювання всіх запитів спрямованих до неї. Ядро передбачає 5 рівневий процес оброблення запиту, що працює за патерном програмування “Ланцюжок відповідальностей” [50]. Кожен рівень отримує запит, модифікує його відповідно до власних налаштувань захисту, та перенаправляє його до наступного елемента в ланцюжку. Елементи є незалежними, тому є можливість змінювати їх порядок. Наприклад, повідомлення запиту спочатку шифрується, а потім стискається, або навпаки.

На рисунку 2.5 графічно зображено ядро системи з рівнями захисту. Кожен рівень – це окрема лінія. Оскільки рівні є незалежними, їх можна вмикати та вимикати незалежно від інших рівнів. Червовий колір лінії означає, що її рівень вимкнений, зелений – безпека на даному рівні ввімкнена.

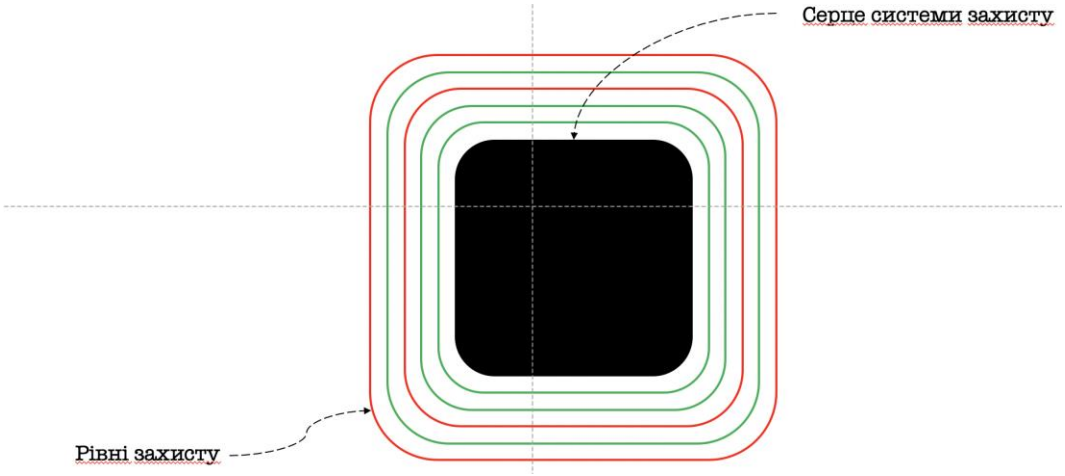

Рисунок 2.5 – Ядро системи з рівнями захисту

Рівень бази даних складається з одного серверу бази даних типу H2 (рисунок 2.6) та схеми по-замовчуванню. H2 – це швидка база даних, що працює в оперативній пам’яті. Під час закриття програмного продукту, вся інформація та відповідні зміни зберігаються до спеціального файлу відновлення. Шлях до файлу можна налаштувати.

Також, присутня опція не зберігати дані. Тобто, відключення серверу безпеки призведе до втрати всіх змін зроблених після останнього запуску сервера. Це означає, що наступний запуск системи захисту відновить стандартні налаштування безпеки.

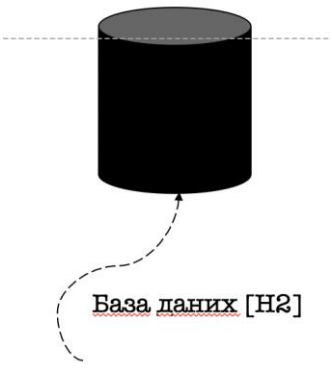

Рисунок 2.6 – Використана база даних H2

2.3 Архітектура налаштування системи безпеки

Архітектура налаштування системи безпеки реалізована як REST (англ. Representational State Transfer) система на основі протоколу HTTPS та з hypermedia-driven архітектурою.

Сервер з підтримкою hypermedia-driven архітектури надає інформацію для навігації по REST русурсам динамічно, включаючи посилання до різних пов’язаних джерел. Ця можливість відрізняється від можливостей систем на базі SOA та інтерфейсів, керованих WSDL.

Працювати з hypermedia-driven системою дуже просто, оскільки вона разом з необхідною інформацією повертає також пов’язану корисну інформацію в вигляді посилань на REST ресурси.

Для звернення до налаштування системи захисту можна використовувати будь-який HTTP-клієнт, що дозволяє відправляти тіло та конфігурувати заголовки.

На рисунку 2.7 продемонстровано запит на початкову сторінку конфігурації.

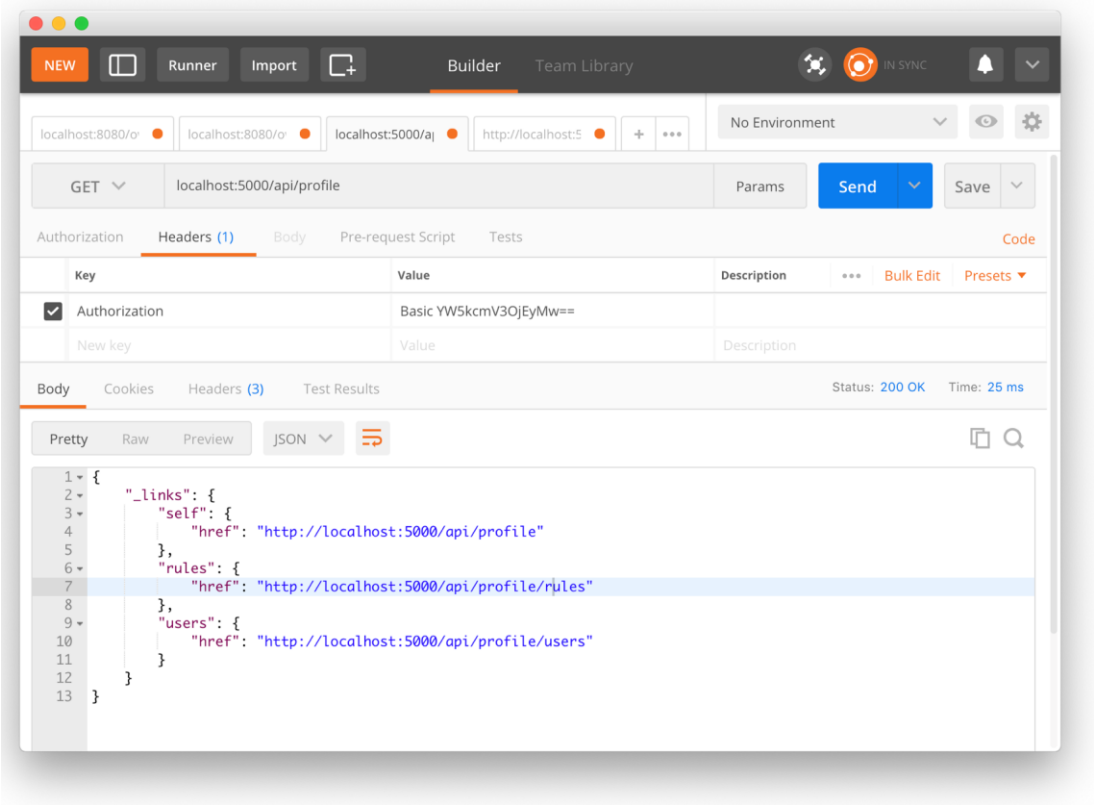

Рисунок 2.7 – Початок конфігурування системи захисту

Відповідь містить посилання на наступні сторінки конфігурації системи. Як приклад, на рисунку 2.8 зображено запит для отримання користувачів системи.

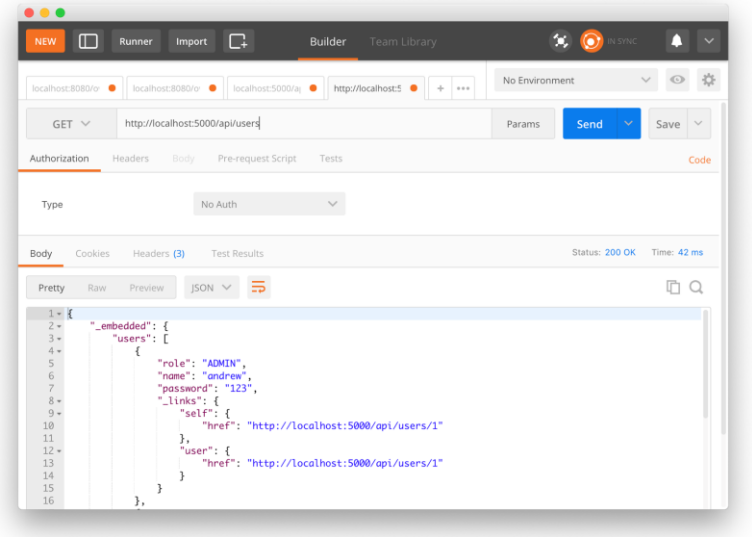

Рисунок 2.8 – Перегляд користувачів системи

Користувач вибирає потрібне посилання, відправляє запит по цьому посиланню, та отримує наступні кроки, які необхідно зробити.

Висновки до другого розділу

В даному розділі було проаналізовано вибрані технології для побудови системи захисту та було детально розглянута архітектура системи.

Даний розділ допомагає поглибити розуміння призначення програмного продукту, та архітектурних рішень покладених в основу систему захисту та системи конфігурування цієї системи.

3. МЕТОДИКА РОБОТИ КОРИСТУВАЧА З СИСТЕМОЮ

Розглянемо клієнтську частину програмного продукту. Спочатку вивчимо інтерфейс користувача, потім дослідимо як система реагує на некоректні дії.

3.1 Інтерфейс користувача

Інтерфейс системи можна умовно поділити на три частини.

Зверху розміщена частина спілкування з сервером системи моделювання гідроакустичних процесів. Це панель, що відображає графічні елементи для віддаленої взаємодії з іншою системою. Під основними елементами управління знаходиться вбудоване вікно для отримання повідомлення від різних зовнішніх джерел, що якимось чином контактують з даним продуктом.

Середньої частиною графічного інтерфейсу є глобальне керування безпекою підопічної системи. Тобто, в цьому місці розміщено кнопку для повного ввімкнення або повного вимкнення системи безпеки та передбачено механізми для автентифікації користувача. На рисунку 3.1 зображено початковий стан системи при першому вході.

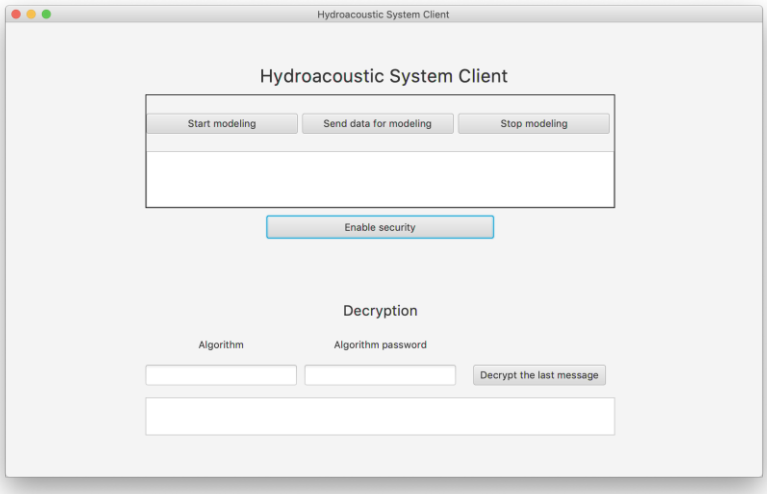

Рисунок 3.1 – Початковий стан системи

Нижня частина графічного інтерфейсу відповідає за зручне дешифрування повідомлень отриманих від системи моделювання гідроакустичних процесів. Аналогічно до верхньої панелі, секція дешифрування має вбудоване вікно, що дозволяє відображати розшифровані повідомлення в хронологічному порядку.

Варто зазначити, що в системі реалізовано механізм збереження значень всіх користувацьких дій під час останнього сеансу. Ця технологія дозволяє відтворити стан вікна, що був на момент останнього закриття програмного продукту. Для користувача це є чудовим графічним досвідом, оскільки не потрібно вводити інформацію, що вже була введена раніше, та особливо якщо введена інформація не потребує змін.

Розглянемо приклад коли користувач заповнив деякі текстові поля та закрив дану програму. Наступного разу при запуску додатку, користувач отримає смогу побачити попередні значення з останньої сесії. На рисунку 3.2 продемонстровано автоматичне заповнення поля “алгоритм дешифрування” при наступному запуску системи, що було отримано з попереднього сеансу.

Рисунок 3.2 – Приклад автоматичного заповнення поля

Для моделювання гідроакустичного процесу було передбачено основні та найважливіші команди:

- команда, що запускає процес моделювання гідроакустичного процесу;
- команда, що відправляє файл до системи моделювання;
- команда, що зупиняє процес моделювання гідроакустичного процесу.

Для кожної команди даної клієнтської системи передбачена відповідний графічний елемент – кнопка, що відправляє запит на виконання відповідної функції.

Розглянемо приклад, в якому користувач запускає систему моделювання гідроакустичного процесу натискаючи кнопку з верхньої панелі “Почати моделювання”. В разу успішного виконання запиту, вікно повідомлень поповниться новим повідомленням. Стан графічного відображений на рисунку 3.3.

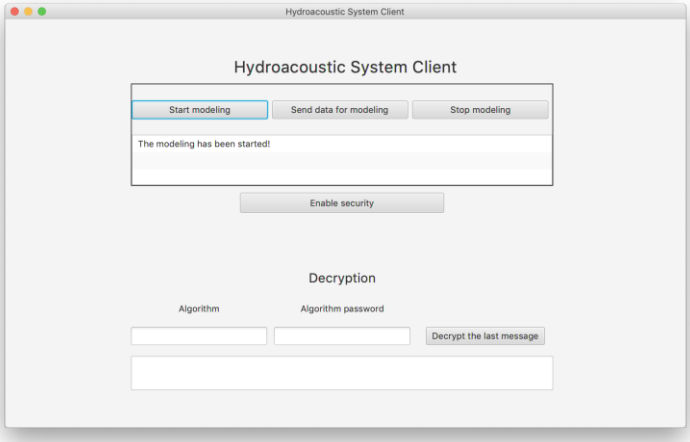

Рисунок 3.3 – Стан системи після запуску процесу моделювання

Наступний приклад – запит на зупинку системи моделювання гідроакустичного процесу. Користувач натискає кнопку “Зупинити моделювання”, що розміщена на верхній панелі та отримає відповідне повідомлення. Стан системи відображено на рисунку 3.4.

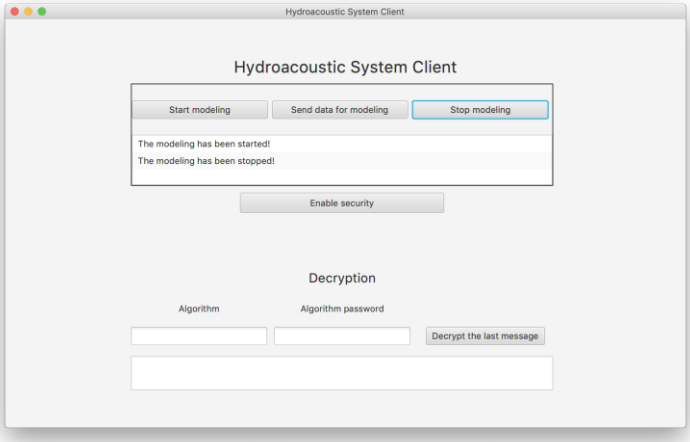

Рисунок 3.4 – Стан системи після зупинки процесу моделювання

Важливим етапом кожного процесу моделювання є надання інформації для моделювання, або її уточнення. Для цього була розроблена кнопка “Відправити дані для моделювання”. Після натиснення на цю кнопку, користувач повинен вибрати файл, що містить потрібну інформацію для моделювання, за допомогою системного вікна, що зображене на рисунку 3.5.

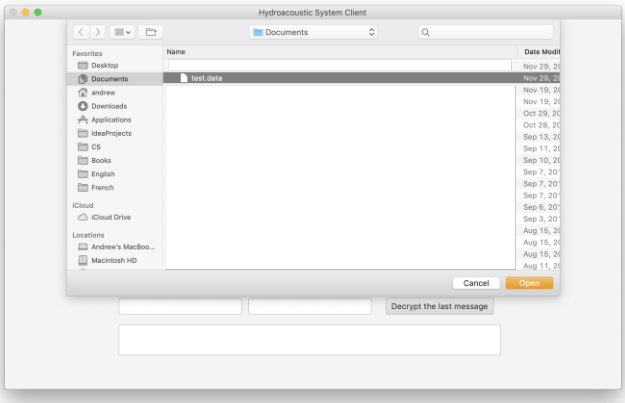

Рисунок 3.5 – Приклад вікна для вибору файлу з даними для моделювання

Варто зауважити, що вигляд вікна для вибору файлу може відрізнятися в різних системах, але його функція залишається незмінною для всіх платформ – дозволити отримати посилання на файл з програми.

Після ініціалізації команди відправлення файлу, користувач повинен отримати повідомлення про результат запиту. В нашому випадку, був відправлений файл з вмістом “test data”. Сервер моделювання успішно отримав файл, прочитав його вміст та відповів клієнтському доданку включивши дані, які він прочитав з отриманого файлу. Даний стан системи відображено на рисунку 3.6.

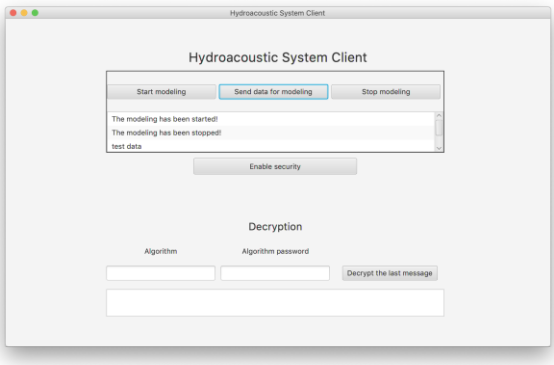

Рисунок 3.6 – Повідомлення про успішне отримання файлу сервером

Розглянуті вище операції описують систему до інтегрування з розробленою системою захисту, а саму сервером-посередником між системою моделювання гідроакустичних процесів та її клієнтами.

Зараз ми перейдемо до функції керування ввімкненням та вимкненням серверу для забезпечення безпеки. Для цього розглянемо кнопку “Ввімкнути безпеку” (при першому вході) в середній частині графічного інтерфейсу. Дана кнопка є динамічною і, залежно від стану, може виконувати дві операції:

- ввімкнення системи безпеки;
- вимкнення системи безпеки.

Написи до кнопки також є динамічними, тобто такими що змінюються після виконання будь-якої операції. Користувач отримає чудовий графічний досвід, оскільки назва кнопки завжди описує дію яку вона виконує. Результат натиснення на кнопку “Ввімкнути безпеку” зображений на рисунку 3.7.

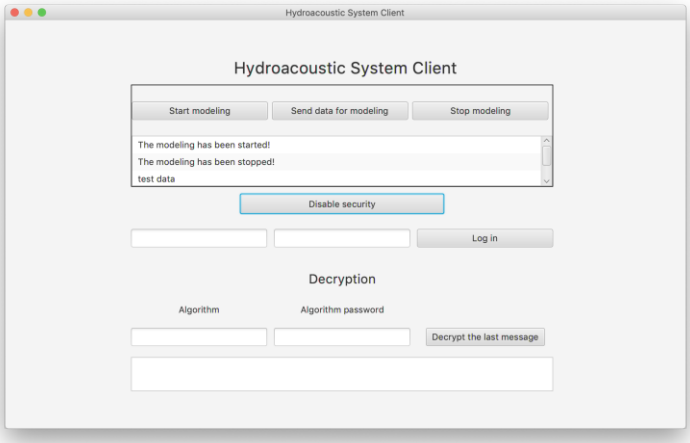

Рисунок 3.7 – Стан системи після натиснення на кнопку “Ввімкнути безпеку”

Як ми можемо спостерігати, динамічна кнопка змінила свою назву на “Вимкнути безпеку”. Це означає, що повторне її натиснення призведе до вимкнення системи-посередника.

Одночасно з ввімкненням системи безпеки, користувачу надається можливість ввійти в систему, що є сервер-посередником, що зараз забезпечує можливості автентифікації та авторизації для системи моделювання гідроакустичних процесів.

Форма представлена в середній зоні графічного інтерфейсу під кнопкою глобального налаштування безпеки. Вона продемонстрована на рисунку 3.8 складається з трьох графічних елементів:

- текстове поле для логіну користувача;
- текстове поле для паролю користувача;
- кнопка для підтвердження проведення автентифікації.

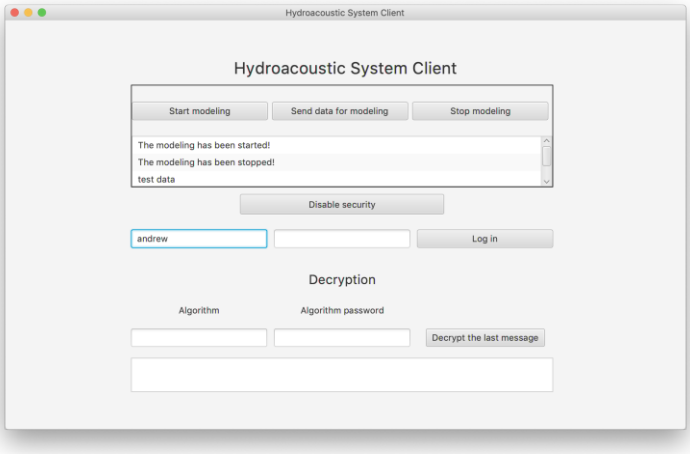

Рисунок 3.8 – Форма для автентифікації

Варто зазначити, що текстове поле для паролю користувача маскує будь-який символ, що друкується, тобто, забезпечує повну конфіденційність на стороні клієнта на його користувача. Приклад введення даних в поле паролю показаний на рисунку 3.9.

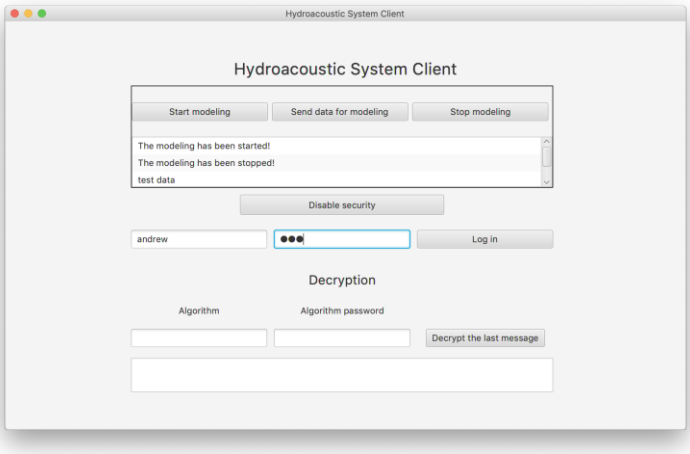

Рисунок 3.9 – Введення даних в поле для паролю користувача

Після успішного введення даних для автентифікації, користувач побачить напис типу “Ви ввійшли в систему як <ім’я користувача>”. Наприклад, для користувача andrew повідомлення буде відформатовано до наступного формату “Ви ввійшли в систему як andrew”. Даний стан системи зображений на рисунку 3.10.

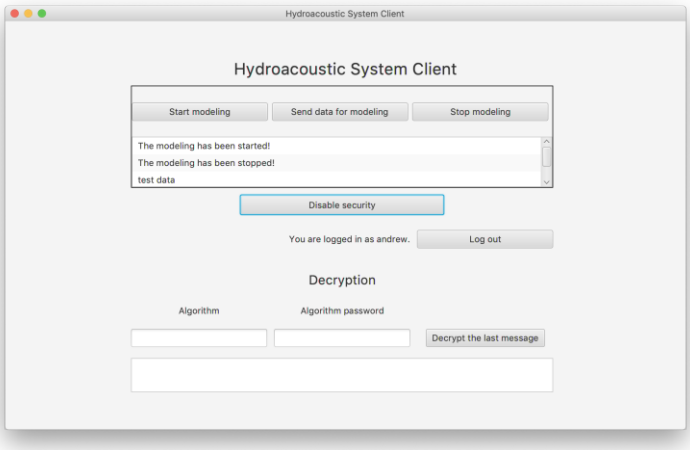

Рисунок 3.10 – Повідомлення про успішний вхід в систему

Звичайно, коли користувач зайшов до системи, він має можливість вийти з неї натиснувши на відповідну кнопку “Вийти”. Наступне отримане повідомлення зображене на рисунку 3.11 свідчитиме про успішний вихід.

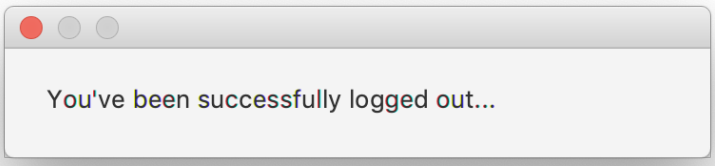

Рисунок 3.11 – Повідомлення про успішний вихід з системи

Важливо зауважити, що вхід в клієнтську систему не перевіряє користувацькі дані на сервері безпеки, оскільки це не має значення до першого запиту клієнта. Вхід в клієнтську систему означає запам’ятовування даних, які потрібно включити при відправці всіх наступних запитів.

При першому з’єднанні сервера з клієнтом, сервер перевіряє відправлені дані і реагує відповідно до налаштувань автентифікації та авторизації. Розглянемо найважливіші помилки, що можуть виникнути в ході цих двох процесів.

По-перше, користувач може ввести недійсний логін, тобто ім’я, що відсутнє серед користувачів системи захисту. Сервер визначить це і поверне повідомлення про помилку, що зображена на рисунку 3.12.

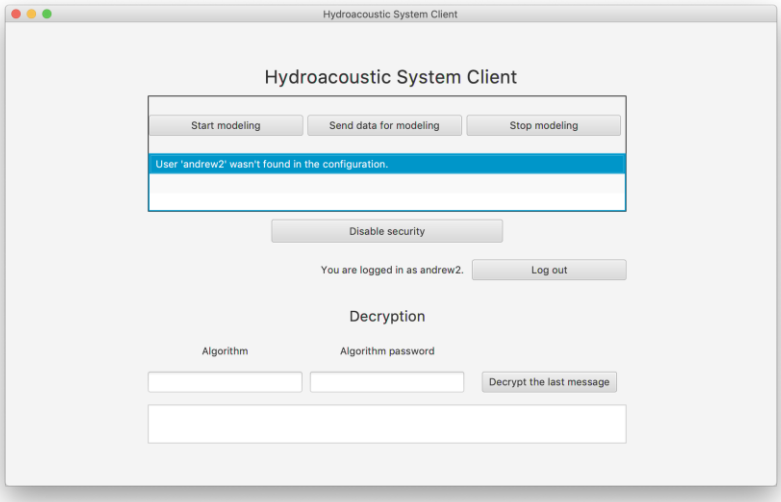

Рисунок 3.12 – Користувач не зареєстрований в системі

Припустимо, що користувач ввів коректне значення для логіну, але помилився з паролем. Сервер може розпізнати це, та повернути повідомлення в форматі, зображеному на рисунку 3.13.

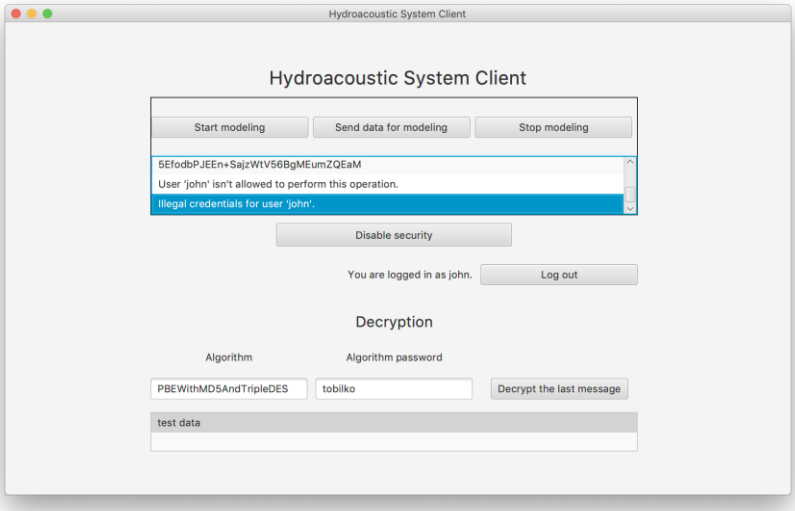

Рисунок 3.13 – Користувач помилився при введенні даних

Припустимо, що користувач успішно пройшов стадію автентифікації. Далі слідує стадія авторизації, тобто етап перевірки повноважень перевіреної особи на серверній стороні и для автентифікованого користувувача перевіряється його набір ролей.

Розглянемо на прикладі користувача john, що успішно автентифікувався на сервері, але має обмежені права для роботи з системою моделювання. Припустимо, що йому не дозволено зупиняти моделювання гідроакустичного процесу.

На рисунку 3.14 зображено повідомлення, що буде надіслано john, якщо він спробує виконати операцію, на виконання якої він не має прав.

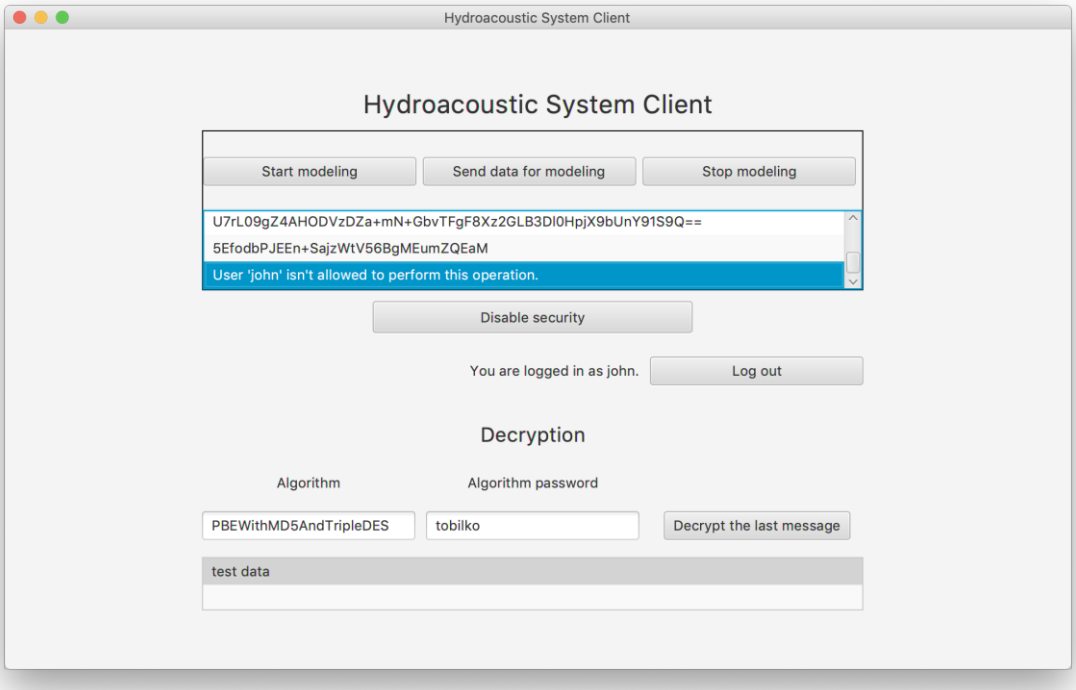

Рисунок 3.14 – Заборона виконання операції

Було розглянуто основні проблеми, що можуть виникнути у користувача на стадіях автентифікації та авторизації.

Зараз, розглянемо стадію шифрування даних та відправимо запит на запуск системи від імені коректного користувача і з ввімкненою функцією безпеки.

Як ми можемо бачити на рисунку 3.15, користувач отримав повідомлення в зашифрованому вигляді.

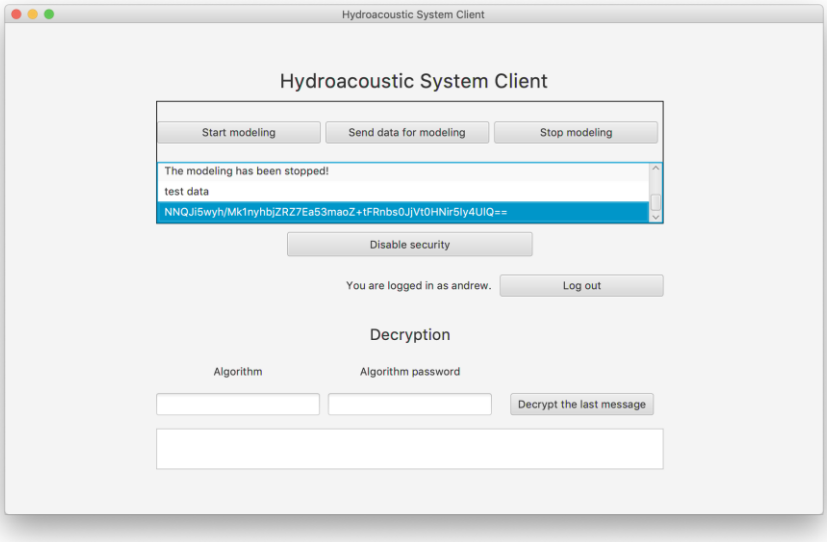

Рисунок 3.15 – Шифроване повідомлення після запиту на запуск системи

Залишивши попередні налаштування, продовжимо звертатися до серверу моделювання в режимі шифрування. Цього разу спробуємо зупинити систему, та отримаємо повідомлення зображене на рисунку 3.16.

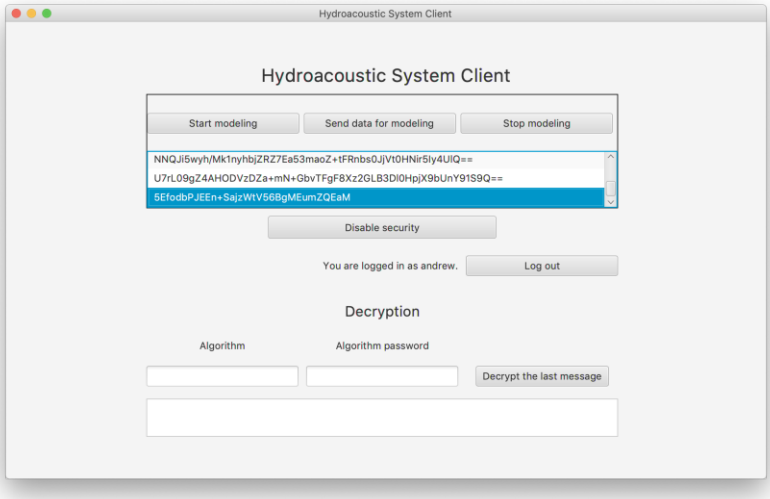

Рисунок 3.16 – Шифроване повідомлення після запиту на зупинку системи

Перевіримо функцію шифрування файлів. Користувач завантажує файл способом описаним раніше та підтверджує свій вибір. Отримане повідомлення також містить зміст файлу, але цього разу в зашифрованому вигляді. На рисунку 3.17 відображено стан системи після отримання відповіді від серверу моделювання.

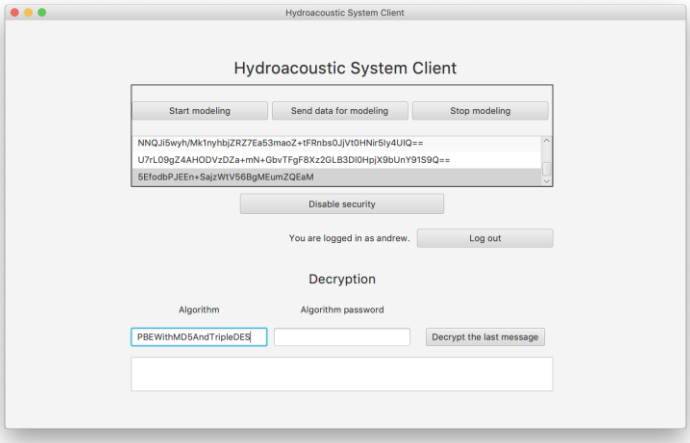

Рисунок 3.17 – Шифроване зміст відправленого файлу

Шифрування має велику цінність для забезпечення безпеки системи в цілому. Хоча, з практичної точки зору, клієнтський додаток зазвичай повинен мати змогу розшифрувати повідомлення отримані від системи моделювання гідроакустичних процесів.

Важливо було забезпечити швидкий та зручний інтерфейс для дешифрування даних, аналогічно до можливості шифрування. Функція “Дешифрування”, що зображена на рисунку 3.18, розміщена знизу та має три основні елементи управління:

- текстове поле для назви алгоритму шифрування;
- текстове поле для ключа шифрування;
- кнопка для виклику функції дешифрування з вказаними параметрами.

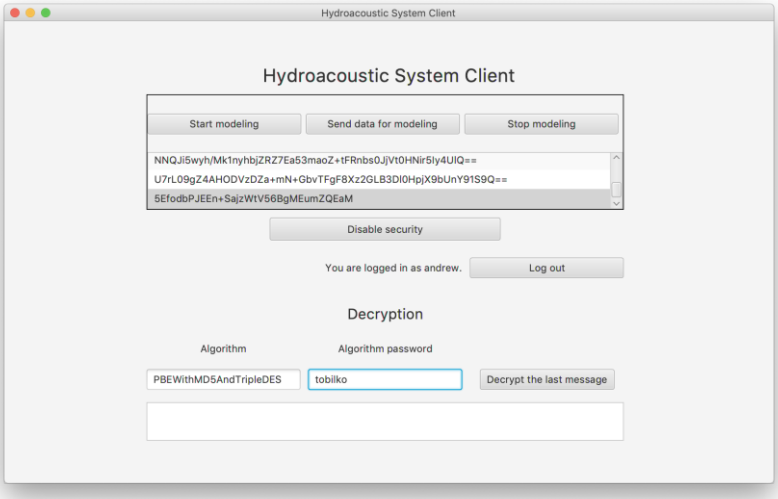

Рисунок 3.18 – Функція дешифрування

Крім елементів управління, інтерфейс має вбудоване вікно для нотифікацій. Користувач має змогу переглянути дешифровані повідомлення в цьому вікні, як це зображено на рисунку 3.19.

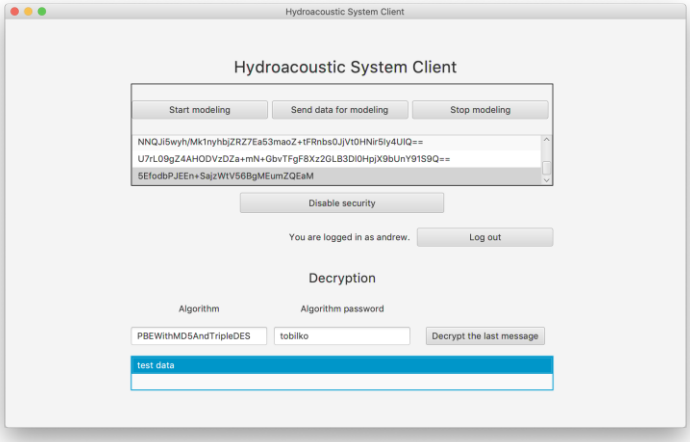

Рисунок 3.19 – Перегляд дешифрованого повідомлення

3.2 Реагування системи на некоректні дії користувача

Клієнтська частина програмного продукту передбачає реагування на некоректні дії користувача при взаємодії з графічним інтерфейсом. Перебачено велику кількість повідомлень та нотифікацій згідно до дії користувача.

Розглянемо кілька типових прикладів.

В формі введення даних для автентифікації перебачено два поля текстових поля: логін користувача та його пароль. Якщо користувач вводити некоректні дані необхідні для логіну, він отримає повідомлення типу, що зображено на рисунку 3.20.

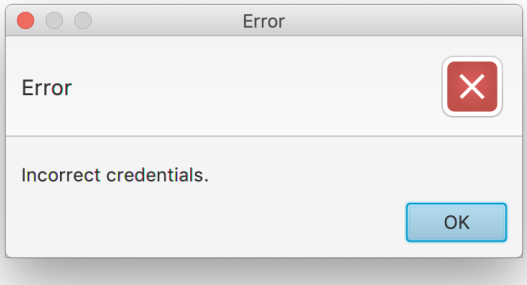

Рисунок 3.20 Приклад повідомлення про помилку при автентифікації

Наступним яскравим прикладом може бути приклад валідування полів для отримання дешифрованого повідомлення. В нижній частині клієнтського додатку є поля для введення алгоритму шифрування (англ. algorithm name) та ключа алгоритму (англ. algorithm password або algorithm key). Система реагує на некоректні дані, введені в ці поля та красномовно повідомляє користувача. Помилкою може бути некоректна назву алгоритму, або назва алгоритму, що не підтримується системою. Помилкою також вважається некоректне значення ключа (рисунок 3.21).

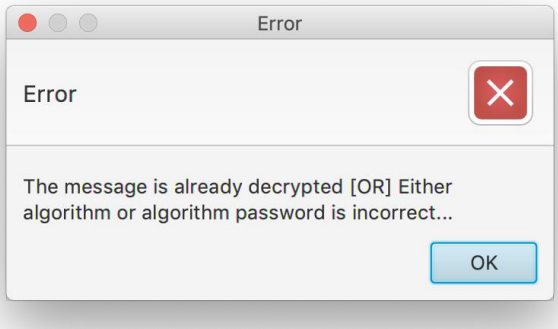

Рисунок 3.21 Приклад повідомлення про помилку при дешифруванні

Висновки до третього розділу

Даний розділ надає детальну інструкцію користувача, послідовно та зрозуміло пояснює кожну особливість графічного інтерфейсу.

Цей розділ поглиблює розуміння системи захисту, та надає можливість протестувати її на прикладі тестового сервера системи моделювання гідроакустичних процесів та розробленого до неї графічного клієнта.

4. РОЗРОБЛЕННЯ СТАРТАП ПРОЕКТУ

Впродовж останнього десятиліття, стартап проекти набули широкого розповсюдження. Стартап – це форма малого венчурного підприємництва. Популярність стартап проектів зараз зумовлена зниженням бар'єрів виходу на ринок. В появою таких потужних інструментів комунікації, як глобальна мереже Інтернет, стало набагато простіше знаходити споживачів та інвесторів, організовувати пошук необхідних ресурсів, перетинати кордони між ринками різних країн. Старт проекти вважаються однією із головних складових інноваційної економіки. Загальна кількість інноваційних ідей зростає за рахунок гнучкості та мобільності стартап-проектів, що дозволяє розвивати економіку країни.

Варто зауважити, що створення та впровадження стартап-проекту відзначається ризиком. Як демонструє статистика, лише невелика частина проектів стає успішними, за різними оцінками це значення складає 10% – 20%. Варто розуміти, що ідея стартап-проекту є важливою складовою, але без працюючої бізнес-моделі та без чіткого формування концепції послуги чи товару вона не буде успішною. На початковому етапі формування концепції та бізнес-моделі є головним завданням керівника проекту.

Для того щоб стартап проект потрапив на ринок, існує певна кількість етапів, що визначають ринкові перспективи проекту, графік та принципи організації виробництва, фінансовий аналіз та аналіз ризиків і заходи з просування пропозиції для інвесторів.

Розглянемо маркетинговий аналіз стартап проекту. В межах цього етапу:

- розробляється опис самої ідеї проекту та визначаються загальні напрями використання потенційного товару чи послуги, а також їх відмінність від конкурентів;
- аналізуються ринкові можливості щодо його реалізації;
- на базі аналізу ринкового середовища розробляється стратегія ринкового впровадження потенційного товару в межах проекту.

4.1 Опис ідеї проекту

В межах даного підрозділу були проаналізовані і подані у вигляді таблиць наступні питання:

- зміст ідеї (що пропонується);
- напрямки застосування;
- основні вигоди, що може отримати користувач товару чи послуги;
- відмінності від існуючих товарів чи послуг.

Перші три питання розглянуті у вигляді таблиці (таблиця 4.1) і дають цілісне уявлення про зміст ідеї та можливі базові потенційні ринки, в межах яких потрібно шукати групи потенційних клієнтів.

Таблиця 4.1 — Опис ідеї стартап-проекту

Зміст ідеї	**Напрямки застосування**	**Вигоди для користувача**
Надання механізмів захисту для системи моделювання гідроакустичних процесів	1. Системи моделювання гідроакустичних процесів	1. Система захищена за принципами 5 рівнів
	2. Системи, що потребують захисту даних з якими працюють	
	3.Системи, що працюють у рамках HTTP протокола	

Аналіз потенційних техніко-економічних переваг ідеї порівняно із пропозиціями конкурентів передбачає:

- визначення переліку техніко-економічних властивостей та характеристик ідеї;
- визначення попереднього кола конкурентів (проектів-конкурентів) або товарів-замінників чи товарів-аналогів, що вже існують на ринку, та проводиться збір

інформації щодо значень техніко-економічних показників для ідеї власного проекту та проектів-конкурентів відповідно до визначеного вище переліку;

- проводиться порівняльний аналіз показників: для власної ідеї визначаються показники, що мають а) гірші значення (слабкі сторони); б) аналогічні (нейтральні сторони) значення; в) кращі значення (сильні сторони) (таблиця 4.2).

Таблиця 4.2 - Визначення сильних, слабких та нейтральних характеристик ідеї проекту

No		(потенційні) товари/концепції конкурентів		
		Мій проект	Система безпеки 1	Система безпеки 2
1	Слабка сторона	Робота лише з HTTP/HTTPS протоколами	Підтримка кількох протоколів	Підтримка SOAP архітектурних рішень
2	Сильна сторона	Проста та прозора інтеграція до вже існуючих систем	Вбудовані механізми безпеки, що можна конфігурувати	Вбудовані механізми безпеки, що не можна змінити
3	Нейтральна сторона	Налаштування системи безпеки здійснюється через API	Налаштування системи безпеки здійснюється через графічний інтерфейс	Налаштування системи безпеки не здійснюється

Визначений перелік слабких, сильних та нейтральних характеристик та властивостей ідеї потенційного товару є підґрунтям для формування його конкурентоспроможності.

4.2 Технологічний аудит ідеї проекту

В межах даного підрозділу було проведено аудит технології, за допомогою якої можна реалізувати ідею проекту (технології створення товару). Визначення технологічної здійсненності ідеї проекту передбачає аналіз таких складових (таблиця 4.3):

- технологія виготовлення товару згідно ідеї проекту;
- наявність потреби допрацювання технології;
- доступність техгологій.

Таблиця 4.3 — Технологічна здійсненність ідеї проекту

No	**Ідея проекту**	**Технології її реалізації**	**Наявність технології**	**Доступність технології**
1	Інтерфейс користувача	Мова програмування Java	Наявна	Умовна безкоштовно
2	База даних	H2	Наявна	Умовна безкоштовно
3	Алгоритми захисту системи	Мова програмування Java	Відсутня	Відсутня
4	Алгоритми написання серверу	Мова програмування Java	Відсутня	Відсутня
Висновок: проект реалізувати можливо за вибраними технологіями.				

За результатами аналізу таблиці зроблено висновок щодо можливості технологічної реалізації проекту. Технологічним шляхом реалізації проекту було обрано такі технології, як Java, JavaFX, H2 через їх доступність та безкоштовність.

4.3 Аналіз ринкових можливостей запуску стартап-проекту

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, які можуть перешкодити реалізації проекту, дозволяє спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів-конкурентів.

Місткість ринку товарів, коло споживачів яких доволі широке, може бути визначена за допомогою статистичних методів, що враховують як тенденції минулих років у збуті товарів, так і перспективні (фактори науково-технічного прогресу, їх динаміку).

Спочатку було проведено аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (таблиця 4.4).

Таблиця 4.4 — Попередня характеристика потенційного ринку стартап-проекту

No	**Показники стану ринку**	**Характеристика**
1	Кількість головних гравців, од	3
2	Загальний обсяг продаж, грн/ум.од	300 грн
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає
6	Середня норма рентабельності в галузі (або по ринку), %	50 %

Середню норму рентабельності в галузі було порівняно із банківським відсотком на вкладення. Останній є меншим, тому є сенс вкладати гроші саме у цей проект.

За результатами аналізу (таблиця 4.4) було зроблено висновок, що ринок є привабливим для входження.

Надалі були визначені потенційні групи клієнтів, їх характеристики та сформовано орієнтовний перелік вимог до товару для кожної групи (таблиця 4.5).

Потенційний клієнт - термін, що означає приватну особу або організацію, які не є в даний момент клієнтом компанії, але що входять в цільову ринкову групу цієї компанії. Теоретично, при якісній роботі, всі потенційні клієнти можуть стати реальними, якщо будуть виконані певні умови (більш інтенсивна реклама, підвищення якості, зниження ціни, вміла робота менеджерів з продажу).

Таблиця 4.5 — Характеристика потенційних клієнтів стартап-проекту

No	**Потреба, що формує ринок**	**Цільова аудиторія**	**Відмінності у поведінці різних потенційних цільових груп клієнтів**	**Вимоги споживачів до товару**
1	Адаптація методів захисту систем моделювання	Гідроакустичні системи	Компанії заключають довготривалі договори, а стартапери – пробний термін	стабільність роботи; невисока ціна; наявність випробувального періоду; наявність документації; підтримка необхідних платформ

Після визначення потенційних груп клієнтів було проведено аналіз ринкового середовища: складено таблиці факторів, що сприяють ринковому впровадженню проекту, та факторів, що йому перешкоджають (таблиця 4.6, 4.7).

Таблиця 4.6 - Фактори загроз

No	**Фактор**	**Зміст загрози**	**Можлива реакція компанії**
1	Підходить для нових проектів	Потребує визначеної структури бази даних	Імпорт схеми бази даних
2	Власний формат зберігання	При необхідності потрібно розробка сервісу преведення до визначенного формату	Додавання можливості автоматизованого експорту в різні типи сховищ, розробка додаткового ПЗ
3	Обмеженість функцій	Інструмент обмежений наявними функціями і не має деяких функцій, які мають конкуренти	Додавання нових функцій за потреби

Пропозиція - це обсяг товарів та послуг, який виробники хочуть і можуть поставити на ринок за різною ціною за певний проміжок часу.

Сталий причинно-наслідковий зв'язок між ціною та обсягом товарів (послуг), який товаровиробник здатний поставити на ринок, виражається законом пропозиції.

Сталий причинно-наслідковий зв'язок між ціною та обсягом товарів (послуг), який товаровиробник здатний поставити на ринок, виражається законом пропозиції.

Таблиця 4.7 - Фактори можливостей

No	**Фактор**	**Зміст можливості**	**Можлива реакція компанії**
1	Незалежність від платформи	Можна використовувати як web інтерфейс, так і мобільний	Вихід на мобільний ринок, вихід на рівень web додатків
2	Недоліки в існуючих альтернативах	Існуючі альтернативи або працюють повільно, або не є орієнтованими на конкретну предметну область	Модифікація існуючих платформ

Надалі було проведено аналіз пропозиції: визначили загальні риси конкуренції на ринку (таблиця 4.8).

Пропозиція - це обсяг товарів та послуг, який виробники хочуть і можуть поставити на ринок за різною ціною за певний проміжок часу. Сталий причинно-наслідковий зв'язок між ціною та обсягом товарів (послуг), який товаровиробник здатний поставити на ринок, виражається законом пропозиції. Сталий причинно-наслідковий зв'язок між ціною та обсягом товарів (послуг), який товаровиробник здатний поставити на ринок, виражається законом пропозиції.

Закон пропозиції — при інших незмінних чинниках величина (об'єм) пропозиції збільшується у міру збільшення ціни на товар.

Зростання величини пропозиції товару при збільшенні його ціни обумовлене в загальному випадку тією обставиною, що при незмінних витратах на одиницю товару із збільшенням ціни росте прибуток і виробникові (продавцеві) стає вигідним продати більше товару. Реальна картина на ринку складніша за цю просту схему, але виражена у ній тенденція має місце.

Таблиця 4.8 - Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	**В чому проявляється дана характеристика**	**Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)**
1. Вказати тип конкуренції - монополія	На ринку присутні декілька компаній-конкурентів, але їх товар дещо відрізняється між собою.	Підтримка якості продукту та постійні нововведення, вдосконалення.
2. За рівнем конкурентної боротьби - міжнародний	Компанії-конкуренти з інших країн	Створити основу ПП таким чином, щоб можна було переробити даний ПП для використання у галузях інших країн.
3. За галузевою ознакою - міжгалузева	Продукт може використовуватись для різних галузей	Постійне вдосконалення продукту, що не має прив’язки до сфери
4. Конкуренція за видами товарів: - товарно-видова	Конкуренція між видами ПП, їх особливостями.	Створити ПП, враховуючи недоліки конкурентів
5. За характером конкурентних переваг - нецінова	Вдосконалення технології створення ПП, щоб собівартість була нижчою	Удосконалення моделі. Використання більш дешевих технологій для розробки.
6. За інтенсивністю - не марочна	Бренд присутній, але його роль незначна	Реклама, участь у конференціях, семінарах.

Було проведено аналіз конкуренції у галузі за моделлю М. Портера (таблиця 4.9).

Таблиця 4.9 - Аналіз конкуренції в галузі за М. Портером

Складові аналізу	**Прямі конкуренти галузі**	**Потенційні конкуренти**	**Постачальники**	**Клієнти**	**Товари-замінники**
		Системе формування сценаріїв	Мінімізація витрат часу постачальників	Контроль якості	Лояльність споживачів
Висновки:	Визначити інтенсивність конкурентної боротьби з боку прямих конкурентів	Є можливості виходу на ринок, оскільки існуючі рішення не надають потрібних переваг	Постачальники підлаштовуються під ринок	Клієнти диктують вимоги згідно з умовами експлуатації	Обмеження для роботи на ринку через товари замінники

За результатами аналізу таблиця 4.9 було зроблено висновок про можливість роботи на ринку з огляду на конкурентну ситуацію.

Цей висновок був врахований при формулюванні переліку факторів конкурентоспроможності у наступному пункті.

На основі аналізу конкуренції, проведеного в таблиці 4.9, а також із урахуванням характеристик ідеї проекту (таблиця 4.2), вимог споживачів до товару (таблиця 4.5) та факторів маркетингового середовища (таблиці 4.6, 4.7) визначається та обґрунтовується перелік факторів конкурентоспроможності.

Аналіз оформлено у таблиці 4.10. Конкурентоспроможність - здатність підприємства створювати, виробляти і продавати товари та послуги, цінові й нецінові якості яких привабливіші, ніж в аналогічної продукції конкурентів.

Таблиця 4.10 - Обґрунтування факторів конкурентоспроможності

No	**Фактор конкуренто-спроможності**	**Обґрунтування (наведення чинників, що роблять фактор для порівняння конку-рентних проектів значущим)**
1	Орієнтація на предметну область формування сценаріїв розвитку гідроакустики	Існуючі конкуренти або не враховують особливості формування сценаріїв, або виконують процесс побудови не оптимально

За визначеними факторами конкурентоспроможності (таблиця 4.10) проведено аналіз сильних та слабких сторін стартап-проекту (таблиця 4.11).

Таблиця 4.11 - Порівняльний аналіз сильних та слабких сторін проекту

No п/п	**Фактор конкурентоспроможності**	**Бали 1-20**	**Рейтинг товарів-конкурентів у порівнянні з даним продуктом**						
			-3	**-2**	**-1**	**0**	**1**	**2**	**3**
1	Орієнтація на предметну область формування оцінки перспективності територіальних районів для виїзних груп	20	+						

Фінальним етапом ринкового аналізу можливостей впровадження проекту є складання SWOT-аналізу (матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities) (таблиця 4.12).

На основі виділених ринкових загроз та можливостей, та сильних і слабких сторін.

Перелік ринкових загроз та ринкових можливостей було складено на основі аналізу факторів загроз та факторів можливостей маркетингового середовища. Ринкові загрози та ринкові можливості є наслідками (прогнозованими результатами)

впливу факторів, і, на відміну від них, ще не є реалізованими на ринку та мають певну ймовірність здійснення.

Наприклад: зниження доходів потенційних споживачів – фактор загрози.

На основі якого можна зробити прогноз щодо посилення значущості цінового фактору при виборі товару та відповідно, – цінової конкуренції (а це вже – ринкова загроза).

SWOT-аналіз широко застосовується у процесі стратегічного планування, що полягає в розділенні чинників і явищ на чотири категорії.

Таблиця 4.12 - SWOT-аналіз стартап-проекту

Сильні сторони: Актуальність користування системою, яка викликана бажанням розвитку гідроакустики Оцінка проходить відразу для великої кількості людей, а також у будь-який період часу. Актуальність користування системою, яка викликана постійним пошуком співробітників Невелика ціна користування за місяць	**Слабкі сторони:** Потребує масштабної рекламної компан Орієнтація на інтернет, яка може відсіяти «не розвинутих» в технічному плані клієнтів
Можливості: Можливе продовження розробки проекту за кордоном, тому що проблема розвитку електроенергетики актуальна не лише в Україні Систему можна використати на ринку фрілансу, для відсіювання некомпетентних виконувачів	**Загрози:** Відсутність користувачів через погану рекламну компанію

На основі SWOT-аналізу було розроблено альтернативи ринкової поведінки (перелік заходів) для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок.

Визначені альтернативи були проаналізовані з точки зору строків та ймовірності отримання ресурсів (таблиця 4.13).

Таблиця 4.13 - Альтернативи ринкового впровадження стартап-проекту

No	**Альтернатива (орієнтовний комплек заходів) ринкової поведінки**	**Ймовірність отримання ресурсів**	**Строки реалізації**
1	Орієнтація поточної моделі на ринок стартаперів	25 %	8 год
2	Орієнтація поточної моделі на ринок державних установ	20 %	72 год
3	Орієнтація поточної моделі на ринок ентерпрайз	35 %	168 год
4	Переорієнтація на розробку серверної частини	75 %	120 год
5	Переорієнтація на веб-розробку	45 %	96 год

Після аналізу було обрано альтернативу №2.

4.4 Аналіз ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: було проведено опис цільових груп потенційних споживачів (таблиця 4.14).

Таблиця 4.14 - Вибір цільових груп потенційних споживачів

No	**Опис профілю цільової групи потенційних клієнтів**	**Готовність споживачів сприйняти продукт**	**Орієнтовний попит в межах цільової групи (сегменту)**	**Інтенсивність конкуренції в сегменті**	**Простота входу у сегмент**
1	Стартапери	Готові	Високий	Висока	Просто
2	Державні установи	Потребують недовгих переговорів	Середній	Середня	Складно
3	Ентерпрайз	Потребують довгих переговорів	Низький	Низька	Дуже складно
Які цільові групи обрано: стартапери					

За результатами аналізу потенційних груп споживачів було обрано цільові групу, для яких буде запропоновано даний товар, та визначено стратегію охоплення ринку - стратегію стратегію диференційованого маркетингу (компанія працює з декількома сегментами).

Для роботи в обраних сегментах ринку сформовано базову стратегію розвитку (таблиця 4.15).

Таблиця 4.15 - Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	**Стратегія охоплення ринку**	**Ключові конкурентос-проможні позиції відповідно до обраної альтернативи**	**Базова стратегія розвитку**
Орієнтація поточної моделі на ринок стартаперів	Стратегія концентрованого маркетингу	Стартапери потребують швидкості розробки, яку надає підтримка декількох платформ даним продуктом	Стратегія спеціалізації (спирається на диференціацію)

Наступним кроком обрано стратегію конкурентної поведінки (таблиця 4.16).

Таблиця 4.16 - Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку?	**Чи буде компанія шукати нових споживачів**	**Чи буде компанія копіювати основні характеристики конкурента**	**Стратегія конкурентної поведінки**
Ні	Шукати нових споживачів, забирати існуючих у конкурентів		Стратегія заняття конкурентної ніші

На основі вимог споживачів з обраних сегментів до постачальника (стартап-компанії) та до продукту, а також в залежності від обраної базової стратегії розвитку та стратегії конкурентної поведінки розроблено стратегію позиціонування (таблиця

4.17), що полягає у формуванні ринкової позиції (комплексу асоціацій), за яким споживачі мають ідентифікувати торгівельну марку/проект.

Позиціювання — це маркетингове забезпечення товарові бажаного місця на ринку і у свідомості потенційних покупців (образ). Позиція компанії чи продукту показує чим він унікальний УТП (унікальну торговельну пропозицію), чим відрізняється від конкурентів (відстройка від конкурентів), чим корисний споживачу.

Таблиця 4.17 - Визначення стратегії позиціонування

№ п/п	**Вимоги до товару цільової аудиторії**	**Базова стратегія розвитку**	**Ключові конкурентоспроможні позиції власного стартап-проекту**	**Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)**
1	Легкість розуміння, зручний інтерфейс	Стратегія диференціації	Позиція на основі порівняння фірми з товарами конкурентів; Відмінні особливості споживача	Економія часу; Зручність застосування; Практичність та точність результату

Результатом виконання підрозділу стала узгоджена система рішень щодо ринкової поведінки компанії, яка визначає напрями роботи стартап- компанії на ринку.

4.5 Розроблення маркетингової програми стартап-проекту

Сформовано маркетингову концепцію товару, який отримає споживач.

Для цього у таблиці 4.18 підсумовано результати попереднього аналізу конкурентоспроможності товару.

Таблиця 4.18 - Визначення ключових переваг концепції потенційного товару

Потреба	**Вигода, яку пропонує товар**	**Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)**
Пришвидшення оптимальності роботи алгоритму	Побудова оптимального формування сценарію за оптимальний час	Конкуренти або не мають орієнтованості на електроенергетику, або формують сценарії не оптимальним шляхом

Розроблено трирівневу маркетингову модель товару: уточнюється ідея продукту та/або послуги, його фізичні складові, особливості процесу його надання (таблиця 4.18).

Певну популярність отримала модель маркетингу, що складається з трьох рівнів:

- споживачі, що є фундаментом маркетингової діяльності;
- інструменти маркетингу, що включають політику продукту (вироби), політики цін, політику розподілу і політику комунікацій;
- комплекс допоміжних систем, з допомогою яких фірма оцінює фактори, що впливають на її стратегію:

1-й рівень При формуванні задуму товару вирішується питання щодо того, засобом вирішення якої потреби і / або проблеми буде даний товар, яка його основна вигода. Дане питання безпосередньо пов'язаний з формуванням технічного завдання в процесі розробки конструкторської документації на виріб.

2-й рівень Цей рівень являє рішення того, як буде реалізований товар в реальному/ включає в себе якість, властивості, дизайн, упаковку, ціну.

3-й рівень Товар з підкріпленням (супроводом) - додаткові послуги та переваги для споживача, що створюються на основі товару за задумом і товару в реальному виконанні (гарантії якості , доставка, умови оплати та ін).

Таблиця 4.19 - Опис трьох рівнів моделі товару

Рівні товару	**Сутність та складові**		
I. Товар за задумом	Зручність та швидкість отримання практичного результату щодо захисту систем моделювання.		
II. Товар у реальному виконанні	Властивості/характеристики	**М/Нм**	**Вр/Тх /Тл/Е/Ор**
	1. функція побудови моделі захисту		
	Якість: достовірність побудови математичної моделі, достовірність побудови моделі захисту		
	Пакування: відсутнє		
	Марка: StatLabs «Forec»		
III. Товар із підкріпленням	До продажу: відсутнє		
	Після продажу: персональна підтримка в обслуговуванні за додаткову платню.		
Вихідний код та математична модель будуть закриті. На ідею зареєстровано патент.			

Після формування маркетингової моделі товару слід відмітити, що проект буде захищено від копіювання за допомогою ноу-хау.

Наступним кроком є визначення цінових меж, якими необхідно керуватись при встановленні ціни на потенційний товар (остаточне визначення ціни відбувається під час фінансово-економічного аналізу проекту), яке передбачає аналіз ціни на товари-аналоги або товари субститути, а також аналіз рівня доходів цільової групи споживачів (таблиця 4.20).

Аналіз проведено експертним методом.

Таблиця 4.20 - Визначення меж встановлення ціни

N	**Рівень цін на товари-замінники**	**Рівень цін на товари-аналоги**	**Рівень доходів цільової групи споживачів**	**Верхня та нижня межі встановлення ціни на товар/послугу**
1	27… 250 грн	105… 300 грн	25000… 50000 грн	27… 105 грн

Наступним кроком є визначення оптимальної системи збуту, в межах якого було прийняте рішення (таблиця 4.21).

Таблиця 4.21 - Формування системи збуту

No	**Специфіка закупівельної поведінки цільових клієнтів**	**Функції збуту, які має виконувати постачальник товару**	**Глибина каналу збуту**	**Оптимальна система збуту**
1	Клієнт повинен надаватися в режимах “тріал” та “повний” сплатити після закінчення випробувального сроку	Легість в встановленні, легкість в сплаті послуг	Веб-сайт	Проводити збут силами посередника формування сценаріїв

Останньою складової маркетингової програми є розроблення концепції маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів (таблиця 4.22).

Таблиця 4.22 - Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	**Канали комунікацій, якими користуються клієнти**	**Ключові позиції, обрані для позиціонування**	**Завдання рекламного повідомлення**	**Концепція рекламного звернення**
Купляють програми через авторизовану мережу	Веб-сайти	Формування сценарію розвитку	Довести, що продукт оптимально формує сценарій	Формування сценарію розвитку

Результатом підрозділу стала ринкова (маркетингова) програма, що включає в себе концепції товару, збуту, просування та попередній аналіз можливостей ціноутворення, спирається на цінності та потреби потенційних клієнтів, конкурентні переваги ідеї, стан та динаміку ринкового середовища, в межах якого впроваджено проект, та відповідну обрану альтернативу ринкової поведінки.

Висновки до четвертого розділу

В даному розділі було проведено аналіз програмного продукту у якості стартап проекту. Можна зазначити, що у проекту є можливість комерціалізації, оскільки ринок потребує якісний продукт, що надає можливість створювати моделі нелінійних-нестаціонарних процесів.

На ринку наявна монополістична конкуренція, існує декілька фірм-конкурентів, але їх товар дещо відрізняється, тому вихід на ринок не буде легким і потребує грамотної стратегії виходу. Для впровадження ринкової реалізації проекту слід обрати альтернативу, яка передбачає розробку програмного продукту з подальшим розповсюдженням за певну плату.

ВИСНОВКИ

В ході магістерської дисертації було досліджено існуючі системи моделювання гідроакустичних процесів та проаналізовано типові та найважливіші проблеми пов’язані з їх безпекою.

Дослідження базувалося на аналізі системи моделювання на найбільших значимих рівнях, виділеними відомими спеціалістами в галузі комп’ютерної безпеки, серед яких можна виділити наступні:

- рівень автентифікації;
- рівень авторизації;
- рівень шифрування даних;
- рівень балансування навантаження;
- рівень стиснення даних.

Для кожного з перелічених вище рівнів захисту системи було детально проаналізовано проблеми, що можуть виникнути на кожному з них та причини, що їх обумовили.

В результаті дослідження було виявлено набір проблемних питань пов’язаних з безпекою притаманних для типової системи моделювання гідроакустичних процесів.

На основі аналізу предметної області і визначення мети роботи як виявлення небезпек в системі моделювання гідроакустичних процесів та надання механізмів для їх усунення, було сформовано наступні вимоги до програмного рішення. Система безпеки була сформована як така, що:

- гарантує безпеку системи моделювання гідроакустичних процесів;
- надає вищезгадані рівні захисту, з можливістю детального налаштування кожного з рівнів;
- є адаптивним програмним рішенням, тобто таким, що може застосованим до широкої вибірки систем моделювання гідроакустичних процесів.

Результатом виконання магістерської роботи є програмний продукт, що повністю відповідає поставленим задачам та задовільняє всі згадані вимоги. Система безпеки гарантує захист системи моделювання гідроакустичних процесів, надає зручний спосіб інтеграції з підопічною системою, та гнучкий спосіб конфігурування, відповідно до вимог користувача гідроакустичної системи.

Для тестування даного програмного рішення, також була створена тестова система моделювання гідроакустичного процесу, та приклад клієнтського додатку для комунікації з нею. Результатом роботи з гідроакустичних клієнтом є підтвердження якості та надійності роботи системи захисту, що інтегрувалася з основною систем просто та без проблем.

Для налаштування системи захисту був використаний HTTP клієнт POSTMAN, що дозволив детально вивчити API та зконфігурувати систему відповідно до цього інтерфейсу.

Дослідження, що були проведені під час виконання магістерської дисертації, показали, що продемонстрована система захисту має багато інноваційних та перспективних ідеї, на яких у майбутньому потрібно зупинитись та зосередити свої дослідження. Ідея надання системи захисту, що включає рівні безпеки, що можуть бути регульованими під потреби користувача є дуже цікавою, і в майбутньому її можна розвинути до надання більшої кількість таких рівнів, або до створення більш детальних конфігурації для кожного з рівнів. Напрямок робіт в даній галузі вбачається перспективним.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Hydroacoustics: Lakes and Reservoirs [Електронний ресурс]. — Режим доступу: stateofthesalmon.org/fieldprotocols/downloads/SFPH_p5.pdf
2. Web Application Security. A Beginner’s Guide [Електронний ресурс]. — Режим доступу: https://books.google.com.ua/books//w_app_sec.html
3. Емельянов В.В. Теория и практика моделирования / В.В. Емельянов, В.В. Курейчик, В.М. Курейчик. — М: Физматлит, 2003. — 432 с.
4. Underwater Positioning System [Електронний ресурс]. — Режим доступу: http://www.fcrov-piloting.com/company-equipment/usbl-underwater-positioning-system.aspx
5. Документація JetBrains IDEA [Електронний ресурс]. — Режим доступу: https://www.jetbrains.com/idea/docs
6. Гладков Л.А. Защита информационных систем / Л.А. Гладков, В.В. Курейчик, В.М. Курейчик. — 2-е изд. — М.: Физматлит, 2006. — 320 с.
7. Скобцов Ю.А. Основы эволюционных вычислений / Ю.А. Скобцов. — Донецк: ДонНТУ, 2008. — 326 с.
8. "Texas Hospital hacked, affects nearly 30,000 patient records". Healthcare IT News, 2016.
9. Becker, Rachel (27 December 2016). "New cybersecurity guidelines for medical devices tackle evolving threats". The Verge. 2016.
10. "Postmarket Management of Cybersecurity in Medical Devices" (PDF). 2017.
11. Brandt, Jaclyn (2018-06-18). "D.C. distributed energy proposal draws concerns of increased cybersecurity risks". 2018-07-04.
12. Cashell, B., Jackson, W. D., Jickling, M., & Webel, B. (2004). The Economic Impact of Cyber-Attacks. Congressional Research Service, Government and Finance Division. Washington DC: The Library of Congress.
13. Gordon, Lawrence; Loeb, Martin (November 2002). "The Economics of Information Security Investment". ACM Transactions on Information and System Security, 438–457.

14. RFC 2828 Internet Security Glossary
15. CNSS Instruction No. 4009, 2012.
16. "InfosecToday Glossary" (PDF). 2014 [Електронний ресурс]. — Режим доступу: http://www.infosectoday.com/Articles/Glossary.pdf
17. Definitions: IT Security Architecture Archived 15 March 2014 at the Wayback Machine.. SecurityArchitecture.org, Jan, 2006
18. Jannsen, Cory. "Security Architecture". Techopedia. Janalta Interactive Inc., 2014.
19. "Cybersecurity at petabyte scale", 2016.
20. Woodie, Alex (9 May 2016). "Why ONI May Be Our Best Hope for Cyber Security Now", 2016.
21. "Firms lose more to electronic than physical theft". 2015 [Електронний ресурс]. — Режим доступу: https://www.reuters.com/article/2010/10/18/us-crime-fraud-idUSTRE69H25820101018
22. Foreman, P: Vulnerability Management, page 1. Taylor & Francis Group, 2010. ISBN 978-1-4398-0150-5
23. Anna-Maija Juuso and Ari Takanen Unknown Vulnerability Management, Codenomicon whitepaper, October 2010 "Archived copy", 2011.
24. Alan Calder and Geraint Williams. PCI DSS: A Pocket Guide, 3rd Edition, 315-319.
25. Harrison, J. (2003). "Formal verification at Intel". 18th Annual IEEE Symposium of Logic in Computer Science, 2003.
26. Umrigar, Zerksis D.; Pitchumani, Vijay (1983). "Formal verification of a real-time hardware design". ст 221–7.
27. "Abstract Formal Specification of the seL4/ARMv6 API" (PDF), 2015.
28. Christoph Baumann, Bernhard Beckert, Holger Blasum, and Thorsten Bormer Ingredients of Operating System Correctness? Lessons Learned in the Formal Verification of PikeOS, 2011.
29. "Getting it Right" [Електронний ресурс]. — Режим доступу: https://web.archive.org/web/20130504191958/http://www.ganssle.com/rants/gettingitright.htm
30. "National Cyber Security Division". U.S. Department of Homeland Security, 2008.

31. "FAQ: Cyber Security R&D Center". U.S. Department of Homeland Security S&T Directorate. 2008.
32. AFP-JiJi, "U.S. boots up cybersecurity center", 2009.
33. "Federal Bureau of Investigation – Priorities". Federal Bureau of Investigation. 2016.
34. "Internet Crime Complaint Center (IC3) – Home". 2011.
35. "Infragard, Official Site". Infragard. 2010.
36. "Robert S. Mueller, III – InfraGard Interview at the 2005 InfraGard Conference". Infragard (Official Site) – "Media Room". 2009.
37. "CCIPS". Archived from the original on 23 August 2006. [Електронний ресурс]. — Режим доступу: http://www.cybercrime.gov
38. "A Framework for a Vulnerability Disclosure Program for Online Systems". Cybersecurity Unit, Computer Crime & Intellectual Property Section Criminal, 2018.
39. "The History & Future of the U.S. Cyber Command", 2016. [Електронний ресурс]. — Режим доступу: n2information.com.
40. "Speech:". Defense.gov., 2010.
41. Shachtman, Noah. "Military's Cyber Commander Swears: "No Role" in Civilian Networks" Archived 6 November 2010 at the Wayback Machine., 2010.
42. "FCC Cybersecurity". FCC. 2010.
43. "Cybersecurity for Medical Devices and Hospital Networks: FDA Safety Communication". 2016.
44. "Automotive Cybersecurity – National Highway Traffic Safety Administration (NHTSA)". 2016.
45. "U.S. GAO – Air Traffic Control: FAA Needs a More Comprehensive Approach to Address Cybersecurity As Agency Transitions to NextGen". 2016.
46. Aliya Sternstein (4 March 2016). "FAA Working on New Guidelines for Hack-Proof Planes". Nextgov. 2016.
47. Bart Elias. "Protecting Civil Aviation from Cyberattacks" (PDF). 2016.
48. Verton, Dan (28 January 2004). "DHS launches national cyber alert system". Computerworld. IDG. 2008.
49. Clayton, Mark. "The new cyber arms race". The Christian Science Monitor. 2015.

50. Nakashima, Ellen (13 September 2016). "Obama to be urged to split cyberwar command from NSA". 2016.

ДОДАТОК А

Комп’ютерна безпека в системах моделювання гідроакустичних процесів

Апробації

УКР.НТУУ”КПІ ”.ТВ_0000_18М

Аркушів 11

2018

ДОДАТОК Б

Комп’ютерна безпека в системах моделювання гідроакустичних процесів

Акт впровадження

УКР.НТУУ”КПІ ”.ТВ_0000_18М

Аркушів 1

2018